■ PROJECTS ■ THEORY ■ APPLICATIONS ■ CIRCUITS ■ TECHNOLOGY

NUTS AND VOLTS

NUTS AND VOLTS

www.nutsvolts.com
July 2013

EVERYTHING FOR ELECTRONICS

Remote CAMERA CONTROL Using Bluetooth & iOS

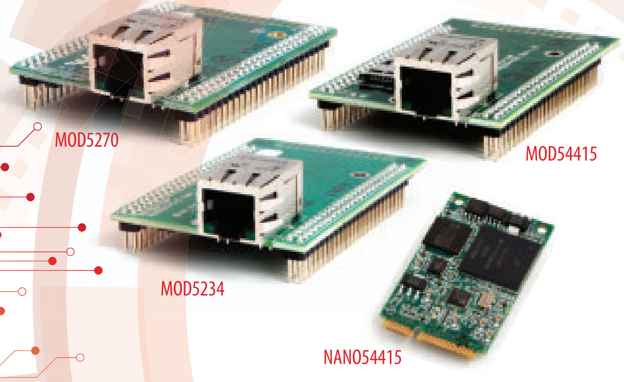
✦ Bringing An Old 1950's Jukebox Amplifier Back To Life

✦ Build An Emergency Radiation Alarm

✦ Make A Hi-Tech Smart Necklace

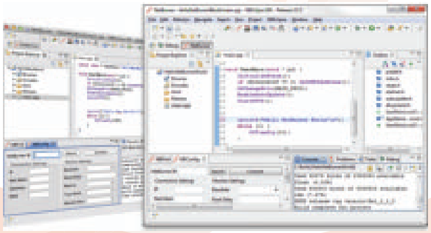
✦ Near Space - LED Photometers

Ethernet Core Modules with
High-Performance Connectivity Options



- **MOD5270**
  147.5 MHz processor with 512KB Flash & 8MB RAM · 47 GPIO
  3 UARTs · I²C · SPI
- **MOD5234**
  147.5 MHz processor with 2MB flash & 8MB RAM · 49 GPIO · 3 UARTs
  I²C · SPI · CAN · eTPU (for I/O handling, serial communications, motor/timing/engine control applications)
- **MOD54415**
  250 MHz processor with 32MB flash & 64MB RAM · 44 GPIO · 8 UARTs
  5 I²C · 3 SPI · 2 CAN · SSI · 8 ADC · 2 DAC · 8 PWM · 1-Wire® interface
- **NANO54415**
  250 MHz processor with 8MB flash & 64MB RAM · 30 GPIO · 8 UARTs
  4 I²C · 3 SPI · 2 CAN · SSI · 6 ADC · 2 DAC · 8 PWM · 1-Wire® interface

Add Ethernet connectivity to an existing product, or use it as your product's core processor



The goal: Control, configure, or monitor a device using Ethernet

The method: Create and deploy applications from your Mac or Windows PC. Get hands-on familiarity with the NetBurner platform by studying, building, and modifying source code examples.

The result: Access device from the Internet or a local area network (LAN)

The NetBurner Ethernet Core Module is a device containing everything needed for design engineers to add network control and to monitor a company's communications assets. For a very low price point, this module solves the problem of network-enabling devices with 10/100 Ethernet, including those requiring digital, analog and serial control.

MOD5270-100IR........$69 (qty. 100)	NNDK-MOD5270LC-KIT..............$99
MOD5234-100IR........$99 (qty. 100)	NNDK-MOD5234LC-KIT............$249
MOD54415-100IR.....$89 (qty. 100)	NNDK-MOD54415LC-KIT..........$129
NANO54415-200IR...$69 (qty. 100)	NNDK-NANO54415-KIT...............$99

NetBurner Development Kits are available to customize any aspect of operation including web pages, data filtering, or custom network applications. The kits include all the hardware and software you need to build your embedded application.

- **For additional information please visit**
  http://www.netburner.com/kits

NetBurner
Networking in One Day!

Information and Sales | sales@netburner.com
Web | www.netburner.com
Telephone | 1-800-695-6828

freescale™
Alliance Member

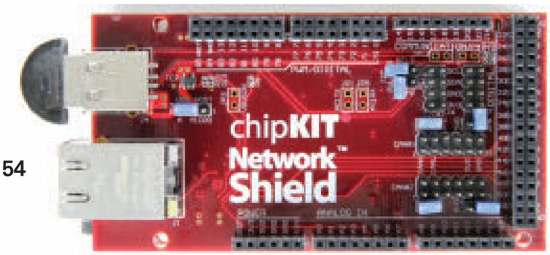
July 2013

BUILD IT YOURSELF

Page 34

PROJECT SPOTLIGHT

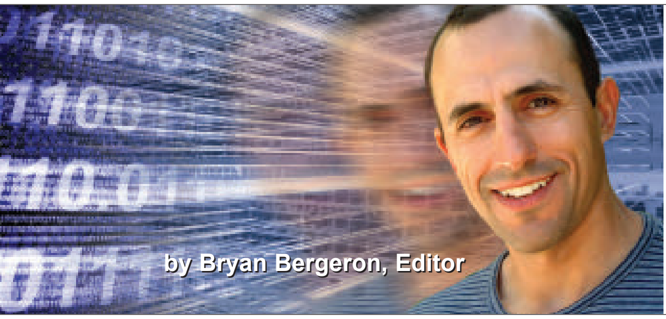
Columns



Page 54

Departments

Nuts & Volts (ISSN 1528-9885/CDN Pub Agree #40702530) is published monthly for $26.95 per year by T & L Publications, Inc., 430 Princeland Court, Corona, CA 92879. PERIODICALS POSTAGE PAID AT CORONA, CA AND AT ADDITIONAL MAILING OFFICES. POSTMASTER: Send address changes to **Nuts & Volts, P.O. Box 15277, North Hollywood, CA 91615** or Station A, P.O. Box 54, Windsor ON N9A 6J5; cpcreturns@nutsvolts.com.



by Bryan Bergeron, Editor

DEVELOPING PERSPECTIVES

Li-ion Battery Technology

A week after the Boeing Dreamliner fleet was grounded due to a lithium-ion battery fire, I attempted to mail a quadcopter to my nephew via the US Postal Service. The counter person at the post office said that I couldn't ship a lithium-ion battery unless it was installed in the quadcopter. Because there was no power switch on the copter — as is normally the case — leaving the battery connected would exhaust and permanently damage the battery. I opted to mail the quadcopter without the 4,700 mAh battery.

Lithium-ion batteries are amazing. Their energy density is significantly greater than that of carbon-zinc, alkaline, or NiMH batteries. For this reason, they're the standard energy source for cell phones, laptops, tablet computers, electric cars, and, most recently, Dreamliners. When energy density is a must, then a li-ion battery may be the only practical option. For example, my fleet of three quadcopters simply wouldn't make it off the ground with bulkier and heavier lead-acid or NiMH batteries.

After investing in a smart li-ion charger — check out **towerhobbies.com** and **xheli.com** — I converted my floor sharks (carpet roamers and crawler robots) to compact, lightweight 3.7V li-ion cells. I've had good luck with SparkFun's li-ion USB charger and 850 mAh battery combination. If you opt for a low voltage overhaul, then you'll have to consider replacing 5V microcontrollers and sensors with 3.3V versions.

Although most consumer li-ion batteries include built-in protection against over-voltage, over-current, and minimum voltage, this protection doesn't guarantee safety. Most of the fires and explosions associated with li-ion batteries — whether in laptops, cars, or planes — occur during the charging cycle. Lithium-ion batteries are prone to thermal runaway. In this positive feedback scenario, when cells overheat due to over-charging, the elevated temperature speeds the chemical reaction in the cell which releases more energy as heat, and so on.

The best-case scenario — which typically occurs with smaller, low-capacity li-ion cells — is for the battery to exhaust itself and cool down. Other possible outcomes range from a slow, controlled meltdown of the battery to a more spectacular explosion accompanied by flames.

Given the potential for smoke damage and fire, I had to modify my casual routine of dropping NiCd or NiMh cells into a charger and then checking on them hours later. Now, I charge the cells on a metal desktop, in plain sight, and only when I'm working in my shop. In addition, when I'm charging a battery larger than about 4 Ah, I put the battery in a fireproof safety sack (about $5). Finally, I store the cells in a metal drawer with ventilation holes in the sides and back.

I admit that I'm a little more casual with the micro li-ion batteries rated at less than 100 mAh. Even so, I store them in the same metal drawer that I use for the large bricks.

The bottom line is that when working with li-ion, you should respect the tradeoff of safety versus increased energy density. If and when you decide to make the move to li-ion, invest in a good charger. Or, better yet, build one. If you have a good bench DC power supply or even a wall wart, then all you need is a smart regulator circuit — an excellent weekend project. **NV**

READER FEEDBACK

Shedding Light On A Misstatement

As I was reading "Getting Started with Matrix LED Displays" in the May 2013 issue of *Nuts & Volts*, I noticed a statement that is incorrect and misleading.

The final two sentences in the "Basic Configuration" section are: "Just make sure that only one column is chosen at a time. Otherwise, a row select line would be asked to supply more than the requisite 10 mA, putting the microcontroller at risk."

While it is true that one should not enable more than one column at a time, damaging the microcontroller is NOT the reason. If two or more columns are on at a time, the load on the Row Select driver is still a 330 ohm resistor in series with an LED drop and a Vce(sat) to ground. The current from the Row Select driver remains unchanged (to a first order approximation). The problem is that now this current is divided among two or more LEDs instead of all going

NUTS AND VOLTS
EVERYTHING FOR ELECTRONICS

Published Monthly By
T & L Publications, Inc.
430 Princeland Ct.
Corona, CA 92879-1300
(951) 371-8497
FAX **(951) 371-3052**
Webstore orders only **1-800-783-4624**
www.nutsvolts.com

Subscriptions
Toll Free **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277
North Hollywood, CA 91615

FOUNDER/ASSOCIATE PUBLISHER
Jack Lemieux

PUBLISHER
Larry Lemieux
publisher@nutsvolts.com

**ASSOCIATE PUBLISHER/
VP OF SALES/MARKETING**
Robin Lemieux
display@nutsvolts.com

EDITOR
Bryan Bergeron
techedit-nutsvolts@yahoo.com

CONTRIBUTING EDITORS
Jeff Eckert, Russ Kincaid
Joe Pardue, Fred Eady
Jon McPhalen, Paul Verhage
Rodney Colton, Ron Newton
Michael Wieckowski, Craig Lindley

CIRCULATION DEPARTMENT
subscribe@nutsvolts.com

SHOW COORDINATOR
Audrey Lemieux

**MARKETING COORDINATOR
WEBSTORE**
Brian Kirkpatrick
sales@nutsvolts.com

WEB CONTENT
Michael Kaudze
website@nutsvolts.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION
Sean Lemieux

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *Nuts & Volts Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *Nuts & Volts*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *Nuts & Volts*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: 430 Princeland Court, Corona, CA 92879.

Printed in the USA on SFI & FSC stock.

through one LED. If the LEDs are matched, the current will divide equally and all will be dimmer. If the LEDs are not matched, one LED will "hog" more of the current and the LEDs will have different brightness levels.

This is why only one column should be selected at a time. It "looks bad" and is bad engineering practice, but will not damage the circuit.

Steve Marum

Many thanks for your comments. Yes, my mind must have been in neutral while writing that line. I'm usually thinking about protecting the PIC while interfacing it to outboard stuff. In this case, as you correctly note, that's not the issue here. A person should avoid selecting more than one column merely so the LEDs illuminate properly.

Thanks for catching that.

Thomas Henry

Shredding Tips

In the article "How to Make Professional Looking Panels and Enclosures" from the May 2013 issue, Gordon McComb identified a problem of shredding (similar to forming burrs in metals) when drilling. Here are a few ideas that may help get rid of "shredding:"

(1) Be sure to secure the part to be drilled to keep it from rotating and "walking" up the drill.

(2) Be sure the drill is perpendicular to the paper before drilling.

(3) Turn the piece over and very lightly press the drill into the finished hole to trim the burrs (a $10 deburring tool works on metal, but I am not sure it is needed for or would work on a paper substrate).

**Tim Brown PhD, PE
Honea Path, SC**

Feeling Vulnerable

This is in response to the May 2013 editorial by Bryan Bergeron on "Vulnerabilities of Networked, Embedded Systems." I am a retired elec/mech engineer with experience in programmable control, including microprocessors, etc.

It is the height of stupidity to design and build any machine that can be subject to intrusion from outside sources. No instrumentality should be readable, programmable, or controllable from hackers.

Equipment makers want to sell exotic stuff, but the user is supposed to be smart enough to control his future absolutely and not to depend on fate for anything. I recognized this years ago when this remote control mania started. I designed a PLC for a job in Alabama and the managers wanted to access the control remotely for maintenance and troubleshooting only. [The engineering company was in Chicago.]

I designed a wire access but I inserted a pluggable link that I instructed to be removed at all times unless it was needed.

For important public services to be subject to hackers is a betrayal of public trust and should be stopped or reversed if already in place. Even orders to local operators to respond to remote orders should be subject to verification before action is taken.

It is now proven that strong measures have to be taken so as not to cause harm to our society from our enemies. I would urge you to preach isolation from the Internet as I would do if I had a public forum.

Robert Gibson (subscriber)

Thanks for your feedback, Robert and your firsthand account of the problem. I'll add your comments to the Reader Feedback section.

*Bryan Bergeron
NV Editor*

Love That Pi

I have been a subscriber to *Nuts & Volts* for as long as I can remember. It's a great magazine; my subscription is paid up until June 2015.

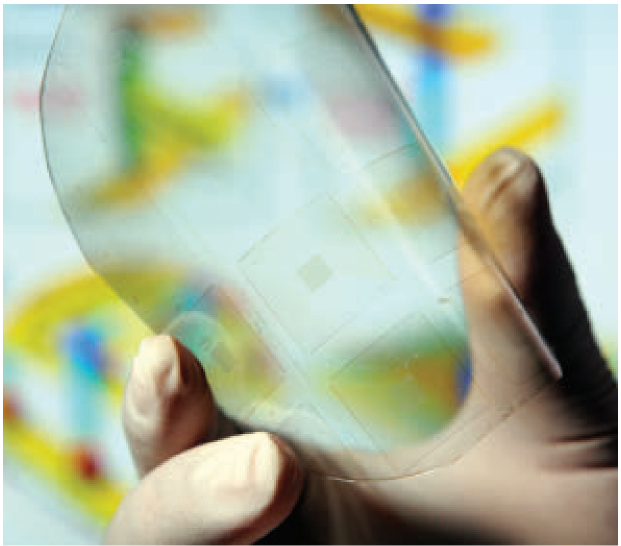
I just wanted to send a note on how much I enjoyed Craig Lindley's article on the Raspberry Pi in the March 2013 issue, from the beginning to the "Resources" at the finish. It's just very well done. I sure hope he does more articles on the

Continued on page 52

■ BY JEFF ECKERT

Discuss these topics at **http://forum.nutsvolts.com**.

ADVANCED TECHNOLOGY



■ Piezotronic transistor array converts mechanical motion into electronic controlling signals.

RAISING OUR TAXELS

There has long been great interest in adding tactile sensing to robots and other devices, and several technologies exist to accomplish it — including a range of capacitive, barometric, and piezoelectric devices. Now, however, researchers at Georgia Tech (**www.gatech.edu**) have developed a new approach to the latter technique. Using bundles of vertically aligned zinc oxide nanowires, they have created arrays of more than 8,000 piezotronic transistors — each of which responds to mechanical strain by producing a signal that can be used for electronic control. Called "taxels," the devices are essentially two-terminal transistors that — rather than using a gate terminal to control current flow — employ a "strain gating" technique that makes use of charges generated at the Schottky contact interface by the piezoelectric effect when the nanowires are placed under mechanical strain.

According to Prof. Zhong Lin Wang, "Any mechanical motion, such as the movement of arms or the fingers of a robot, could be translated to a control signal. This could make artificial skin smarter and more like the human skin. It would allow the skin to feel activity on the surface." Because the arrays are transparent, flexible, and foldable, they could also be used for such things as fingerprinting systems, devices that emulate the mechanoreceptors in hair follicles or hairs in the cochlea, or shape-adaptive sensing in prosthetic skin. According to Wang, future work will be directed toward producing the arrays from single nanowires instead of bundles, and integrating them into CMOS silicon chips. The use of single wires could improve the device's sensitivity by "at least three orders of magnitude," according to the professor. ▲

CHEAP, FLEXIBLE TEGs

The thermoelectric effect (i.e., conversion of temperature differences into electricity) is anything but new, having been discovered in 1821 by German physicist Thomas J. Seebeck. Hence, it is often referred to as the "Seebeck effect." Until fairly recently, thermoelectric generators (TEGs) have been used primarily in military and space applications (including the Curiosity rover), in automobiles to harvest power from exhaust heat, and in obvious places such as wood stoves, boilers, and other industrial processes. The implementation of TEGs has been limited by their relatively low efficiency weight and cost. Researchers at the Fraunhofer Institute for Material and Beam Technology (**www.iws.fraunhofer.de**) have now devised a way to use the soon-to-be-ubiquitous 3D printer to manufacture TEGs that can be cheaply produced in the form of large-area flexible components and made from nontoxic synthetic materials. These units are especially well suited for use in power station cooling towers.

According to Dr. Aljoscha Roch, "Thermoelectric generators currently have an efficiency of around eight percent. That sounds very small, but if we succeed in producing TEGs cost-effectively on a large scale and from flexible materials, we can install them extensively on the insides of concave cooling tower walls. In this way, through the enormous amount of energy produced in the huge plants — around 1,500 liters of water evaporate per minute — we could generate large quantities of electricity." Dr. Roch is quick to add that the devices are not limited to power plants. "In principle, waste heat is produced through the operation of every technical installation.

With TEG fitted on industrial production lines, in sewerage systems, at large computer centers, or on any type of exhaust air system, very large and hitherto unused sources of energy could be developed." BMW, Ford, and other automakers have been experimenting with various heat recovery systems, so the Fraunhofer devices may end up giving us all a mileage boost. ▲

Photo courtesy of Fraunhofer IWS Dresden.

■ A 3D printed TEG wrapped around a sample component.

COMPUTERS and NETWORKING

IBM ADDRESS THE IoT

■ IBM's MessageSight network appliance can be installed in less than 30 min.

By now, nearly everyone has heard the term "Internet of Things (IoT)," which — like "cloud computing" — is vague enough for marketing purposes and catchy enough that people want it even if they don't know exactly what it is. When you get down to the nitty gritty, however, what we're talking about is a conglomeration of machines, sensors, computers, phones, and other apparatus that — according to IMS Research — will number more than 22 billion web-connected devices by 2020 — most of them mobile. According to IBM, the world is already creating 2.5 quintillion bytes of data every day through our use of social media, digital photos and videos, purchase transactions, GPS signals, and so on. So the question is, "How can we handle all of that traffic?"

Well, a consortium called the Organization for the Advancement of Structured Information (OASIS, **www.oasis-open.org**) is dealing with it via a standard called the Message Queuing Telemetry Transport Protocol (MQTT). It was only a few months ago (March 26th) that the first MQTT Technical Committee meeting was held, but somehow IBM (**www.ibm.com**) already has an MQTT appliance for sale.

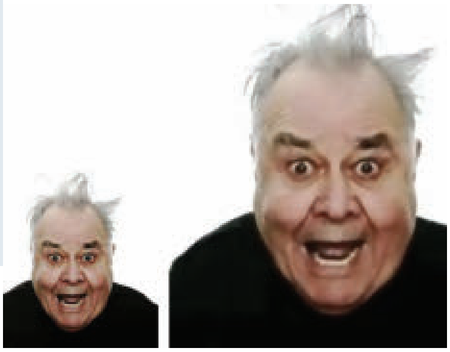
Details about the "MessageSight" box are sketchy as of this writing, and IBM has not even offered a photo of it (we grabbed the image from a video). What is known is that it will enable a communication system to handle data from up to one million devices at a rate of 13 million messages per second. Oddly enough, IBM says that it can be installed and operating in less than 30 minutes, it requires no user-level operating system, and it can be added securely to existing enterprise messaging systems.

What does that mean to us? Well, let's say you're driving around in a car and your iPhone's microphone picks up the sound of your stomach growling. That information could be relayed to a MessageSight box at McDonald's Oak Brook, IL, headquarters, which responds by sending a picture of a Big Mac to your iPad and telling your GPS to provide directions to the nearest restaurant. The possibilities are endless, really. ▲

FREE IMAGE ENHANCEMENT

Somewhere within the University of Texas at Austin (**www.utexas.edu**) lies the Center for Perceptual Systems, where a couple of professors have been working to understand how the visual brain works. In the process, they came to the conclusion that "many tasks the eye performs are the same as what photography requires" — a realization that eventually led to the development of image enhancement software tools.

Unlike other such tools, the employed algorithms are based on the analysis of thousands of images rather than just the one you might be working with. The result is an image enhancement process that can accurately determine what is "noise" in a photo and eliminate it. The profs also claim that it can be used to enlarge images without the loss of quality. The best part is that it is available to everyone for free. All you have to do is log onto **rcm.cps.utexas.edu** and upload up to 1,000 photos per day. Just choose how much enlargement and noise reduction you want, view the result, and download it when you're satisfied. ▲



■ The late, great Jonathan Winters, before and after enlargement/denoise process.

COMPUTERS and NETWORKING continued

WEIRDEST APP YET?

By all accounts, Iceland is a pretty nice place, known primarily for Icelandic sagas, strange foods (they use nets to go "sky fishing" for puffins), an unpronounceable language, and a common belief in the existence of elves. However, Icelandic dating presents some problems that most people don't encounter. First of all, when you are born, your father's first name becomes your last name, so it is nearly impossible to tell if you are related to someone you happen to meet in a local pub. Second, there are only 320,000 Icelanders — most of whom live near the capital, Reykjavík — so there's a good chance that you actually are related to that someone you happen to meet in a local pub. Obviously, if you're not careful and get spurred on by a few shots of brennivín, you could wake up the next morning with your cousin.

To remedy the problem of "accidental incest," developers at Sad Engineer Studios (**www.sadengineer.com**) have come up with an Android app called ÍslendingaApp SES. The app taps into an online genealogical database ranging back 1,200 years, and covers public records of more than 720,000 Icelanders. All you and your prospective objet de l'amour have to do is bump your phones together; the app will display an alert if it detects that you share a grandparent.

Apparently, it's a big hit, because more than 5,000 residents downloaded it in the first few days after it was introduced. ▲

CIRCUITS and DEVICES

CELL PHONES CAN'T HIDE ANYMORE

Berkeley Varitronics Systems (**www.bvsystems.com**) provides a range of test and security instruments for the telecommunications industry, including several models of cell phone detectors. The latest is the Manta Ray detector, designed to allow security personnel to locate and seize contraband and unauthorized phones even if they're hidden within otherwise impenetrable materials. The unit works by identifying specific common components found in all cell phones which allows it to "see" units that are on, active, inactive, or even turned off. Manta Ray works through dense materials such as sheetrock and concrete, and it can even see through many metals — including aluminum, brass, and copper.

■ The Manta Ray cell phone detector operating through a brick wall.

Operation is simple enough that it can be used by both trained security personnel and untrained users. An LCD screen and audible beep provides alerts when it detects the possible presence of a cell phone. A standard 9V battery will keep it running continuously for up to three hours. The Manta Ray is manufactured in the USA, which probably accounts for the price of $499. ▲

FUEL CELLS FOR DEVELOPING WORLD

■ Woman in a Nairobi slum tries out a VOTO charger.

As unlikely as it may seem, it appears that people in many parts of the world live in homes without electrical power, running water, or toilets, and yet they own cell phones. Such people may be forced to walk several miles into town to get a battery recharge, which can be pretty unpleasant on a hot, tropical day. Now, Point Source Power (**www.pointsourcepower.com**) has licensed a slew of patents from Berkley Lab and designed a solid oxide fuel cell called "VOTO," that offers "a simple, inexpensive way to provide electricity to the 2.5 billion people in the world who don't get it reliably."

The device is powered by burning whatever biomaterials you happen to use for cooking or heating, including charcoal, wood, or even cow dung. You just stick it into the fire, and as the fuel cell heats up to 700 or 800°C, it charges up the unit's handle. The handle also contains an LED, so it can be used both for phone charging and as a lamp. In technical terms, the VOTO offers USB standard output of 5V at 500 mA max, and an LED output of 25 lumens. A full charge will provide up to 30 hr of light time. The handle will sell for about $17, and the fuel cell for about $7. Each cell will last only three or four months, but Point Source notes that targeted families typically spend $8 to $12 per month on kerosene for their lamps, so this will actually save them money. Commercial release in Kenya is slated for later this year. The company is looking for distributors, so if you live in a desolate, godforsaken region (e.g., Cleveland), this could be your opportunity. Just shoot an inquiry to **info@pointsourcepower.com**. ▲

INDUSTRY and the PROFESSION

HEY, BUDDY ... CAN YOU SPARE $17 BILLION?

Back in 1995, Apple (**www.apple.com**) was widely perceived to be teetering on the brink of disaster, and it reported a fourth quarter loss of $69 million. In response, Standard and Poor's downgraded the company's bonds to junk-bond grade, so the company had to pay 6.5 percent interest to attract investors to cough up a paltry $575 million. The company's recent record-setting $17 billion bond offering — at a measly one percent interest — was immediately gobbled up by investors from all over the globe. In fact, the bond offering attracted $52 billion in offers, so many investors were turned away. Apple intends to use the money to buy back some of its stock, and to return some dividends to stockholders. Why, you might ask, would a company that reportedly has $145 billion in cash reserves want to borrow money? Well, most of the cash is in overseas operations, and bringing it home would trigger substantial US taxes — much more than the one percent interest on the bonds. It's simple arithmetic. **NV**

BY JON MCPHALEN

Sounding Off With WAV Files

As embedded programmers, we spend a lot of time controlling LEDs and other light sources — many times, a blinky light is a means to make our project "speak." The next logical step, then, is audio output. This may be as simple as a "beep" or "boop" generated on an I/O pin, but in a media-saturated, media-savvy world, that's hardly enough anymore. Yes, we can use an external controller like the Rogue uMP3 to play an audio file or — with some fairly simple circuitry and a bit of code — we can play high quality audio files right out of our project. In fact, using the Propeller, we can even play audio while doing other things.

A few years ago, I was working with a friend to create an audio player for light industrial use: theme parks, museums, dark attractions, etc. We initially focused on the ISD17240 as it was new, promised better fidelity than its predecessor, and seemed like it should be easy to use. To make a long, ugly story very short, things with the ISD17240 didn't go as planned (the vendor later revealed a silicon issue that had confounded us). We were in a pickle.

Go to **www.nutsvolts.com/index.php?/magazine/article/july2013_SpinZone** for any additional files and/or downloads associated with this article. You can also discuss this topic at **http://forum.nutsvolts.com**.

At the time, our development was focused around the SX chip. While we both knew and had used the Propeller (especially me), we had not deployed it in a commercial product. Well, this seemed like a good time for a change. Once we decided to dump the ISD17240 and switch our processor to the Propeller, a whole new world opened up. It took a few months of development — especially since we like to work with potential customers — but in the end, we had a really successful product called the AP-16+.

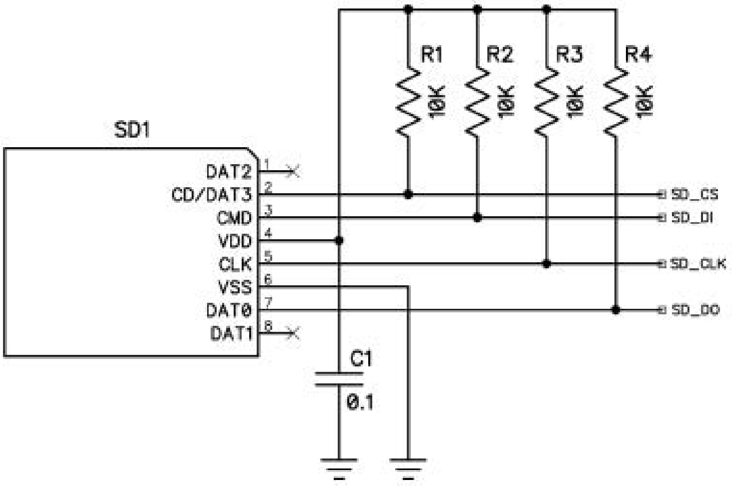
Having the Propeller at our disposal meant that we could read and play WAV files from an SD card. This made loading audio onto the project very easy, and it increased the quality of audio by several orders of magnitude. It also meant that we could update the product by providing a special file to the customer; switching to the Propeller was a big win all around.

"Why WAV files?" you wonder. *"Why not MP3?"* The answer is fairly simple: The MP3 format — which is licensed — was created at a time when storage media (SD cards, etc.) had small capacities and large price tags. That dynamic has flipped: Quality SD cards of gigabytes in size are available for a few bucks. And, frankly, WAVs are easy. There's probably not enough bandwidth in the Propeller 1 to decode MP3 files, and the application would have to be licensed, anyway. Yes, there is MP3 decoder hardware available, but that adds to the cost of materials.

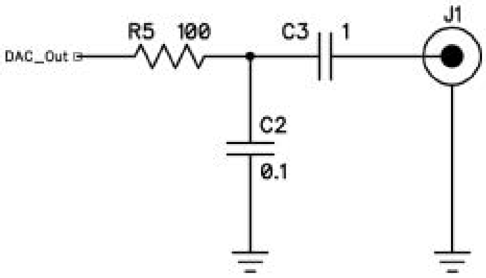
Elements Of A WAV Player

To create a WAV file player with the Propeller, we're going to need the following elements:

- File access hardware and software. I tend to use microSD cards in my personal circuits (see **Figure 1**). For the SD interface, I always use the ***FSRW*** object; it's been around the longest and seems to have the best performance.
- An audio output circuit. For this project, we're going to configure the Propeller's counters as DACs using DUTY mode and an external RC circuit (**Figure 2**) for each channel.
- Finally, the code — this is where we'll spend most of



■ FIGURE 1. SD connections.



■ FIGURE 2. RC DAC connections.

our time as it's the biggest piece of the puzzle. Mind you, it's not very difficult, but it is very involved. In this particular case, we're going to create a complex, multi-cog object that will, in fact, use four of the Propeller's cogs while playing a file.

Elements Of A WAV File

Before we get too far into this, I want us all to remember that no matter how much power we have in the Propeller, it is nothing compared to our PCs — even the run-of-the-mill oldies we may have kept around to run legacy programs.

With that in mind, the code we'll work with here is limited to the *canonical* WAV file format; that is, a simple WAV file with no extra information (metadata).

It is also limited to 16-bit files. Eight-bit files never sound good, anyway, so there's no point in spending time adapting the code to run them. Eight-bit files can always be re-sampled and saved as 16-bit files with an audio editor.

In simple terms, the canonical WAV file has two elements: a header and the audio data — though more technical descriptions will refer to these elements as three "chunks." I strongly suggest that you have a look at this web page: **https://ccrma.stanford.edu/courses/422/projects/WaveFormat/**

Figure 3. WAV Header

Element	Size	Offset
ChunkID	4	0
ChunkSize	4	4
Format	4	8
SubChunk1 ID	4	12
SubChunk1 Size	4	16
Audio Format	2	20
Num Channels	2	22
Sample Rate	4	24
Byte Rate	4	28
Block Align	2	32
Bits Per Sample	2	34
SubChunk2 ID	4	36
SubChunk2 Size	4	40
Data		44

I think this is one of the most useful descriptions of the WAV format on the Internet.

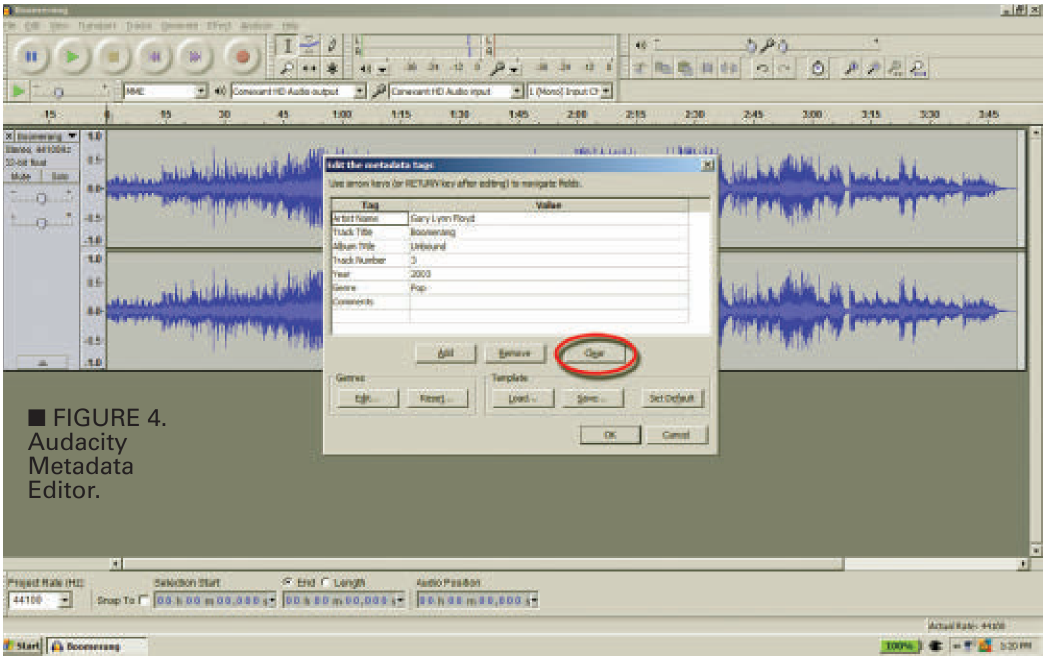
The WAV player object treats the first 44 bytes of the file as the header (**Figure 3** shows the layout of the header data). The header contains technical details required for proper playback of the audio data: number of channels, sample rate, etc. We'll check this section to ensure the file can, in fact, be played before launching the player.

Everything following the header should be audio data. The way these elements are stored make reading and using the Propeller very easy. In a 16-bit stereo file, for example, reading four bytes from the file and stuffing them into a "long" gives us an easy way to transport the left and right channel values through the program.

The player code does allow for mono files; when a file is mono, the player code knows that the long holding samples is actually holding two, and deals with them accordingly.

The Audacity Of Free

One of my favorite freeware programs is Audacity — a nifty little audio creation and editing tool. It runs on any operating system and can open virtually any audio file you throw at it. Not long ago, I received some audio from an



Edit the metadata tags

Use arrow keys (or RETURN key after editing) to navigate fields.

Tag	Value
Artist Name	Gary Lynn Floyd
Track Title	Boomerang
Album Title	Unbound
Track Number	3
Year	2003
Genre	Pop
Comments	

■ FIGURE 4. Audacity Metadata Editor.

Internet radio show in WAV and MP3 format. The creator told me he sent the MP3s because the WAVs were special to his software and I probably wouldn't be able to open them. Ha! Audacity opened the files and allowed me to do what I needed.

What I do most frequently with Audacity is convert MP3s and clean up WAV files for use in a Propeller-powered player. By “clean up,” I mean to remove any non-audio information from the file; this is especially problematic in MP3 files.

Have a look at **Figure 4**. In the image, I've opened an MP3 from friend and extraordinarily talented musician, Gary Lynn Floyd (if you're in Dallas, TX, make sure you get out to one of his shows). After opening the MP3, I clicked on **File\Open Metadata Editor**.

This brings up a dialog that displays the non-audio data included in the file. For a Propeller-powered player, we need to get rid of this information by clicking the **Clear** button.

With that, we can close the Metadata Editor dialog, make adjustments to the file, and then export it as a 16-bit WAV.

Let me suggest that one of the adjustments you make is to normalize the audio to -0.5 dB. This will take the volume up to near maximum without clipping. By normalizing the file like this, our signal-to-noise is increased and we'll get better results from the player. You'll find the Normalize feature in the Effects menu.

Reading And Playing WAVs

First things first: We have to make sure that we have mounted media and can open the target file. As I stated earlier, I use the ***FSRW*** object for file I/O. It's limited to DOS 8.3 filenames, and everything must be in the root menu of the card. For my projects, I don't find these limits problematic — especially as this object works so well.

Back in **Figure 1**, you can see that we need four pins for the SD/microSD card; these get used in the mount methods of ***FSRW***:

```
pri mount | check

  sd.pclose

  check := {
  } \sd.mount_explicit(do, clk, di, cs)

  if (check < OKAY)
    mounted := false
    return NO_MOUNT
  else
    mounted := true
    return OKAY
```

This routine can be called at any point in the program, so it closes any open files first. Then, it connects to the SD card via defined connections. If all goes well (that is, a card was inserted), it will return OKAY. Note that some of the calls to the SD object are preceded with \ which indicates that the method is designed to abort on errors.

FSRW uses zero as the code for "Okay," so that is maintained through the WAV player object. Error codes are expressed as negative numbers, also in keeping with the way ***FSRW*** works.

Before attempting to play a file, we really should check it first — if it's not a proper WAV file, who knows what will come out of the audio side. Most likely, a whole lot of unpleasant noise. We'll start by opening the file and moving the header (first 44 bytes) into an array for analysis:

```
pub get_header(p_filename)

  bytefill(@header, 0, 44)

  if (!mounted)
    mount
    if (!mounted)
      return NO_MOUNT
  if (\sd.popen(p_filename, "r") <> OKAY)
    sd.pclose
    return NO_FILE

  if (\sd.pread(@header, 44) == 44)
    return OKAY
  else
    sd.pclose
    return BAD_HDR
```

This method expects a pointer to a z-string that holds the file name. We can do this inline:

```
  check := get_header(string("jonnymac.wav"))
```

Using the **string** function like this creates a z-string and returns a pointer to it. We can also create a list of files in a DAT section like this:

```
dat

  Jon             byte    "jonnymac.wav", 0
  Intro           byte    "intro.wav", 0
  Winner          byte    "ding.wav", 0
  Loser           byte    "explode.wav", 0
```

I prefer to use the DAT section approach; it makes editing easier when changes are required. Note, though, that when defining file names in this manner we must manually append a 0 as shown.

When using the DAT approach, our call to **get_header** changes to this:

```
check := get_header(@Jon)
```

You'll remember that the @ operator provides a pointer to the element it's attached to.

If the SD isn't mounted, the **get_header** method will remount — if possible — and return an error code if there's a problem (e.g., the SD card was removed). The next step is to open the file in Read mode. Finally, we can use the **pread()** method from ***FSRW*** to move 44 bytes from the file into an array called *header*.

One of the features often missed by new Propeller programmers is the **bytemove** function. This function copies a group of bytes from one place to another.

The first parameter is the address of the destination value/array; the second is the address of the source value/array; and the final parameter is the number of bytes to copy from the source to the destination.

We can use this function to pull elements from the header for testing and direct use.

For example, we should find the characters "WAVE" in the header, starting at offset eight (format field). Part of the file verification is to look for this — here's how we do it:

```
  bytemove(@wavcheck, @header[8], 4)
  wavcheck[4] := 0
  ifnot strcomp(@wavcheck, string("WAVE"))
    return BAD_FILE
```

In this case, we're using **bytemove** to copy four bytes from the *header* (source) array starting at offset eight, to another array called *wavcheck* (destination); this array is terminated with a zero, so we can use **strcomp** to compare two strings (a string is an array of bytes). The program will return an error code if they don't match.

Another use of this technique is to extract values from the array. As I stated, the player is designed for 16-bit WAV files so that should be verified before attempting to play. There is a method in the object just for that:

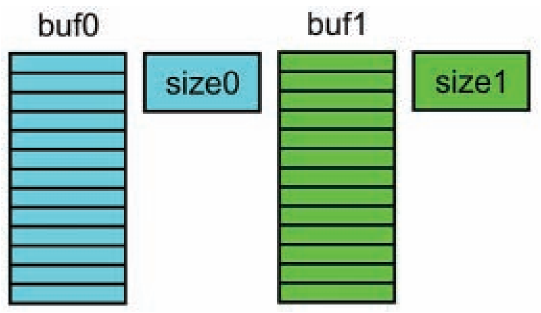
```
pub bits_per_sample | bits

  bits := 0
  bytemove(@bits, @header[34], 2)

  return bits
```

We start by clearing the value in *bits* — this is critical because *bits* is a local variable which means it's a long (four bytes) and being on the stack, could contain anything. With *bits* initialized to zero, we are able to move two bytes from the header — starting at offset 34 — into it using **bytemove**. This works because the values in the header are stored little-endian which is the same way multi-byte values in the Propeller are stored.

Those with experience may wonder why I didn't use **wordmove** here. The reason is that *header* is a byte array; this means it could be aligned on word boundaries in the Propeller memory space, *but there is no guarantee*. The **wordmove** and **longmove** functions require word and long alignment of values. For safety, then, if you're extracting word or long values from a byte array, use **bytemove** as we have here; this ensures success regardless of the alignment of the source array in the Propeller data space.

There are several checkpoints in the object to ensure we can play the file — have a look at the **check_file** method. Let's assume that we've checked a file and it makes the grade. What now? The first thing we have to do is get the data from the file into RAM so we can play it. Ruh-roh ... we have less that 32K, so we're certainly not going to be reading the entire file into the Propeller's RAM.

The good news is that we don't have to. We can read the data in blocks and provide that to the playback element of the code. Honestly, I think this is one of the more interesting and useful aspects of the object, and I have my friend, Peter to thank for it. Peter designed and built a custom animation control system (see **www.socalhalloween.com**) in which he streams animation data via



■ FIGURE 5. Double buffer.

RS-485 to his animatronics. I asked him about the process.

This system uses a double buffer mechanism to provide a constant stream of audio data to the playback cog (which is written in PASM for sample-rate accuracy). It works like this: The program reads as many samples as it can to fill the first buffer, and then writes the number of samples in the buffer to another variable. At the beginning of the file, the value in the size variable will match the size of the buffer. In the player cog, it is waiting for the size value to become non-zero; at that point, it starts extracting the samples and playing them at the rate specified in the WAV file (more on this later).

Figure 5 illustrates the double buffer: For each buffer, there is a separate variable that holds the size (number of samples in the buffer). It is important to fill a buffer before setting the size; another cog will be watching for the size to change from zero and will immediately begin pulling from it.

Back to the "buffer stuffer." After the first buffer is filled, the second is filled, and then the program will loop until the first buffer is empty. The playback cog tells us it's done with a buffer by writing zero to the size variable. This stuff-and-wait process continues until the end of the file.

This buffer stuffing process needs to run in its own cog for us to be able to do other things while the audio is playing. No problem. We can easily launch a Spin method into its own cog so that it runs "in the background" while we do other things. In the AP-16+, for example, the foreground element of the program listens for serial messages from an external controller. This allows the external controller to stop a file in progress or change the volume on-the-fly.

It's a little hairy so I won't get into all of the details, but the actual player cog is necessarily written in PASM so that we can accurately play the file at the desired rate. If our playback rate doesn't match the rate specified in the file, the pitch will be either high (playback is faster than file rate) or low (playback is slower than file rate).

The nice thing about the speed of PASM is that we can, in fact, modify the playback rate and volume while the file is playing. A client once asked for an audio project that would allow him to change the playback pitch and overall volume while the file was playing. This was a fairly straightforward exercise: Read a couple pots with an ADC and then use the **set_speed** and **set_volume** methods to make the desired changes.

I should provide a very strong word of caution here. I tend to use 32 kHz or 44.1 kHz audio files; the lower rate allows speed adjustment on the high side (increased pitch) if I desire. For files that will not require playback rate changes, I stick with 44.1 kHz (CD quality). That said, the object will allow 48 kHz files. I suggest that if you want to play 48 kHz stereo files, you use a 6.25 MHz crystal (for 100 MHz system speed) and a Class 6 or faster SD card. All SD cards are not created equal; check the class rating before buying — especially when using microSD cards in an audio project.

There is one final piece of the firmware puzzle: analog output. In this version of the object, I use a separate cog to update the RC DACs. If you look through the playback code, you'll see that volume-modified samples are reconstructed and then written to the hub like this:

```
playit          mov     arg1, lchan
                mov     arg2, lvol
                call    #adjust
                mov     lchan, arg1

                mov     arg1, rchan
                mov     arg2, rvol
                call    #adjust
                mov     rchan, arg1

                shl     rchan, #16
                or      lchan, rchan
                wrlong  lchan, dacpntr
playit_ret      ret
```

Prior to calling this routine, we have read the left and right samples and volume levels. The call to **adjust** (a modified multiplication routine) sets the channel volume. After both channels have been adjusted, the samples are reconstructed as a long, then written to the hub at the location specified by *dacpntr*.

The logical question becomes, "*Why not write the channel values to the counters which have been set up for analog output?*" That's what I did when first developing this code, but I ran into a very peculiar problem. In many files, the audio would play and be polluted by low level, random noise — kind of a scratching sound, usually at the end of the file. I spent a couple weeks chasing this problem; honestly, I cannot remember how many times I rewrote the code.

My initial thought was that my math routine for adjusting volume was injecting bad results. I wrote a special test program to compare my routine against Spin and could find no differences.

Finally, it was time to call in the "Big Bog." Having known him since 1994, I called the Propeller designer, Chip Gracey and explained what was happening. In less than three seconds he responded, "*I think I know what it is — send me your code*."

It turns out that the interesting architecture of the Propeller creates less-than-interesting audio artifacts when used in this manner. It seems that gate delays across the silicon from the cog driving the counters to the DAC (counter) output pins — coupled with pin updates that may not be in sync — can create clicks, pops, and a bit of scratchy noise when used with a changing analog output (the case with audio).

Now, this noise is not as horrible as I seem to be making it; there are WAV objects available that do not deal with this noise. In my case, however, I was pumping the audio through 20W amps on each channel and the noise was noticeable.

So, the solution? It's as interesting as the problem and reminds me of my friends, Brian Wade and Miyo Nakamura — two amazing sculptors that work for Steve Wang. At the end of each day, they are charged with cleaning the area around the piece they're working on.

Big pieces are sculpted in a material called WED clay, and dried WED clay dust is very fine and extremely difficult to sweep up. Plus, it's bad for the lungs, too. The trick is to pour a solution of a sticky, sawdust-like material onto the clay dust and sweep that up. The clay dust sticks to the sawdust and the result is a clean floor.

So, how exactly, does this result to cleaning the noise out of our audio signal? We clean the noise by adding noise! Yes, this was Chip's solution: Add a controlled amount of noise to the samples.

In the process of launching the WAV player, an object called ***stereo_dacs.spin*** is launched. The purpose of this cog is to configure one or both counters for output; then, sit in a loop reading the sample provided by the player cog, injecting white noise into each sample before writing them to the counters:

```
dacloop      rdlong  lsample, smplspntr
              xor      lsample, Hx80008000

              mov      rsample, lsample
              shl      lsample, #16
              and      rsample, HxFFFF0000

              test     lfsr, taps  wc
              rcl      lfsr, #1

              mov      t1, lfsr
              sar      t1, dadj

              add      lsample, t1
              add      rsample, t1

              mov      frqa, lsample
              mov      frqb, rsample

              jmp      #dacloop
```

As you can see, the major code is short and sweet. The channel samples are read from the hub as a long, converted to unsigned values with XOR, and then split apart. A very simple LFSR routine creates a pseudo-random value which is scaled by the dither depth provided on initialization of this cog. The dither depth adjusts the amount of noise in the signal. The adjusted noise is added to each channel before being written to the counters for output to the RC filter.

Keep in mind that this loop is running many times faster than the normal playback rate, so each sample is modified and written to the DAC outputs many times during the sample rate period. This "whitens" any noise in the signal and allows the RC circuit on the output to remove it. Is it perfect? No, but darned close, and audio experts that have used the AP-16+ have complimented its lack of system noise relative to its [low] price point.

If you're using amplified PC speakers in an application, you will probably never notice. In highly amplified systems, the background noise is no more than would be expected for a typical consumer-level amplifier.

As this is a complex object, let's review the elements and the rolls they play. ***FSRW*** provides file access; it uses a cog for high speed SPI communications with the SD card.

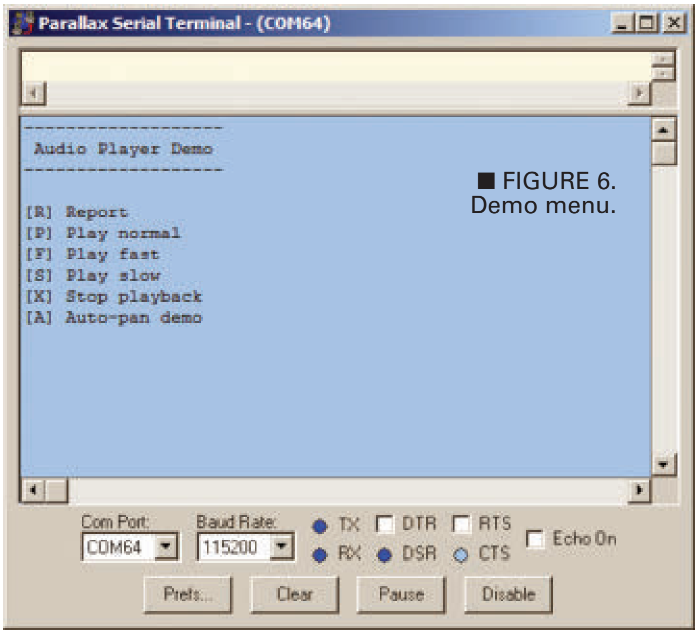
The **wav_spooler** method is launched into a Spin cog to play a file; it uses ***FSRW*** to read the file and uses a double buffer to hold the samples during playback.

The WAV player cog (PASM code is included in the object) reads the samples from the double buffer at the desired playback rate. It also reads the volume and then modifies the left and right channel values accordingly. The left and right channels are reconstructed as a long that is written back to the hub for the ***stereo_dacs*** cog.

Finally, the ***stereo_dacs*** cog reads the volume-adjusted samples, injects a little bit of white noise, then writes the samples to the Propeller counters which have been configured as DACs using DUTY mode and an RC circuit on the respective pins.

If you kept count, you'll see that four cogs are in use while a file is playing. This leaves four cogs for other things — and that's a lot of leftover horsepower.

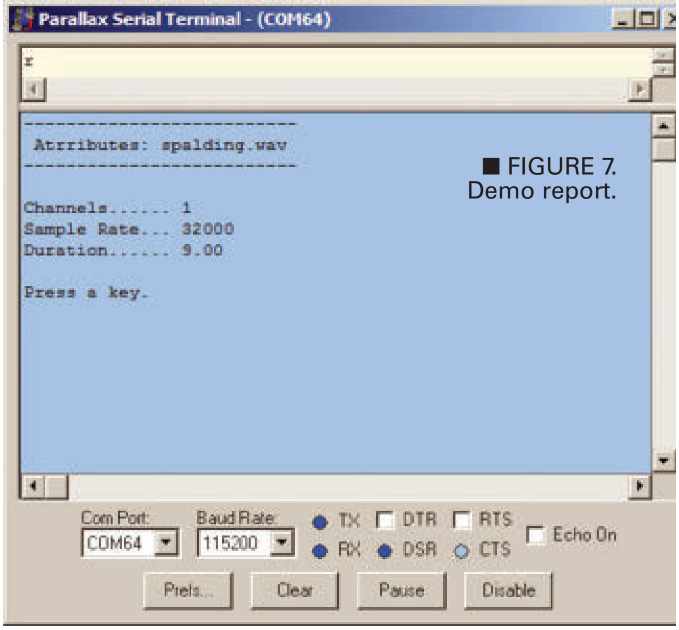
A friend of mine works for a major SoCal amusement park and was tasked with creating a little handheld prop that could play audio files and run a mini light show at the same time. No problem. We used the WAV object and wrote some code that could read light patterns from DAT tables in the program. It was a big hit and deployed to other parks in the company.



```
--------------------
 Audio Player Demo
--------------------

[R] Report
[P] Play normal
[F] Play fast
[S] Play slow
[X] Stop playback
[A] Auto-pan demo
```

Com Port: COM64
Baud Rate: 115200

■ FIGURE 6. Demo menu.



```
r
```

```
--------------------------
 Atrributes: spalding.wav
--------------------------

Channels...... 1
Sample Rate... 32000
Duration...... 9.00

Press a key.
```

Com Port: COM64
Baud Rate: 115200

■ FIGURE 7. Demo report.

Play The WAV

Okay, let's put this dude to use. You'll see that all the code required to enable WAV playback makes the user side of the application very easy. In order to simplify use, all of the cogs described above are loaded via the single object ***jm_wav_player.spin***. This takes care of loading the ***FSRW***, the playback driver, the buffer stuffer, and the ***stereo_dacs*** cog.

To start the object, we need to know the left and right channel pins, the connections to the SD card, and the depth for noise dither used by the ***stereo_dacs*** cog. If you look at the start method of that object, you'll see that it takes a single parameter; this happens to be a pointer to an array of longs that defines the pins and dither setting.

In the demo program, you'll find this:

```
dat

  AudioSetup     long     26, 27
                  long     22, 23, 24, 25
                  long     3
```

The first two numbers are the pins used for the left and right channel outputs. If your setup is only going to use one output (make sure your files are mono), then use –1 in place of a channel number. The second group of numbers are the pins used to connect to the SD card (DO, CLK, DI, CS). The final number is the noise dither depth used by the ***stereo_dacs*** cog. This value will require a bit of experimentation to determine the best setting for your particular setup. A value of one provides the most amount of noise injected into the signal; eight provides the least. Now, we can start the audio object:

```
  audio.start(@AudioSetup)
```

That's it! Now we can put it to use. **Figure 6** shows the menu for a little program that demonstrates the key features of the audio player. Pressing "R" shows the file parameters (**Figure 7**). The **show_report** method opens the file, then uses methods in the WAV player object to provide details about the file.

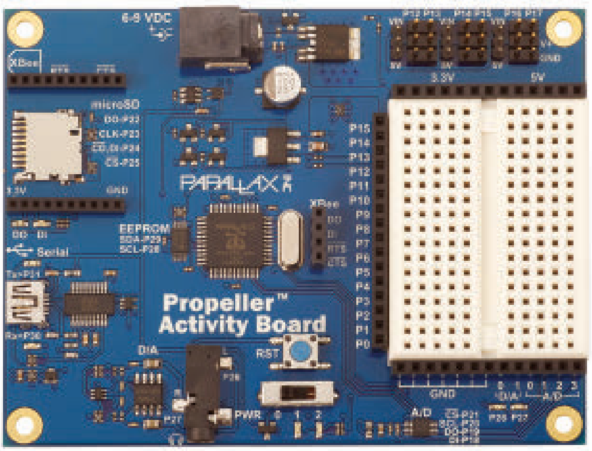
Press a key and return to the main menu. Now, press "P" and the file will play. Remember, playback is controlled by another cog so we can interrupt it. Press the "F" key after a few seconds and you'll hear the file immediately restart using a faster playback rate.

To prove that we can modify playback volume in software while the audio is playing, press the "A" key from the menu. This will start the file playing at 100% volume in the left channel and 0% volume in the right. As the file plays, the audio will pan from the left to the right. By the end of the file, the left output is at 0% volume and the right is at 100% volume. Note that the auto-pan effect is not part of the object. Hence, the way it's written it cannot be interrupted unless coded to do so ("X" will stop the auto-pan demo).

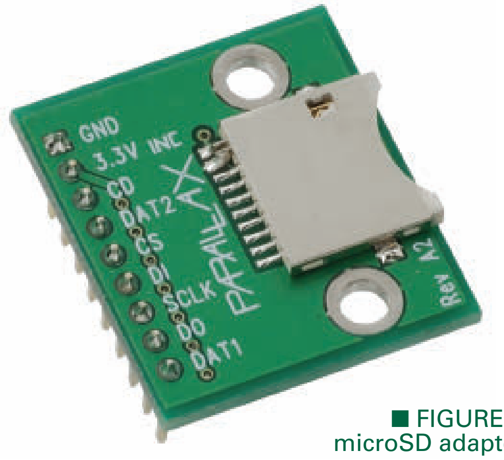
Want to have some fun? Find a laser or "blaster" sound effect on the Internet, save it as a clean 16-bit WAV file,

JON "JONNYMAC" MCPHALEN
jon@jonmcphalen.com

PARALLAX, INC.
Propeller chips and programming tools
www.parallax.com



■ FIGURE 8. Propeller Activity Board.



■ FIGURE 9. microSD adapter.

then run it in the auto-pan demo.

I enjoy Halloween and spend a fair amount of time helping those that build and run haunted houses and other dark attractions. For a place in Knoxville, TN, I wrote a bit of code that would randomly change the playback speed and volume levels of each channel. As humans, we tend to "tune out" repetitive noises and by modifying the pitch (via playback rate) and apparent stereo position, guests in the attraction waiting area are likely to hear the house rules announcement more than once. It was a cool effect, too.

It's time for you to go have some fun now! I used the new Propeller Activity Board (**Figure 8**) while writing my demo code. The Activity Board is a nice little development platform with a great price — just $50. It includes the Propeller, a four-channel ADC, RC DAC circuit and audio output jack, a microSD socket, an XBee socket, servo-compatible headers, a USB programming connection and a small breadboard. Again, it's only $50! This is a great way to get started with the Propeller without spending much money.

Also, Parallax is using the Activity Board as the centerpiece of its C language training for the Propeller. A lot of cool things have been happening in that arena since my last column. Pop over to **learn.parallax.com** for updates.

If you have a Propeller board without an SD socket (like the Propeller PDB), Parallax has a cute little microSD socket (**Figure 9**) adapter that is breadboard-friendly. I've used this adapter with my PPDB in several projects.

Until next time, then, keep spinning, winning, and sounding off with the Propeller! **NV**

In this column, Russ answers questions about all aspects of electronics, including computer hardware, software, circuits, electronic theory, troubleshooting, and anything else of interest to the hobbyist. Feel free to participate with your questions, comments, or suggestions. **Send all questions and comments to: Q&A@nutsvolts.com**

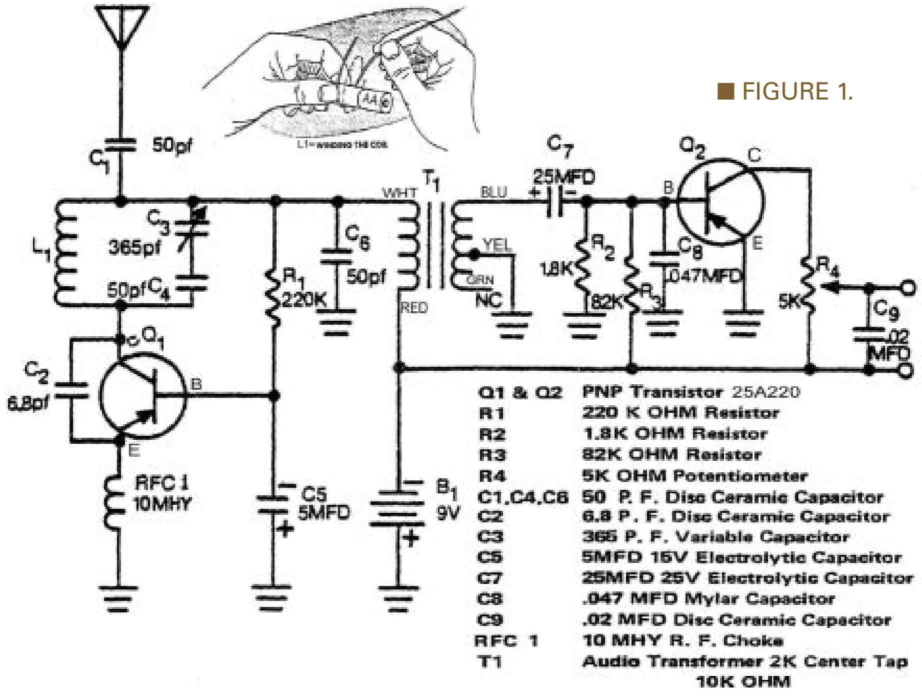
Go to **www.nutsvolts.com/index.php?/magazine /article/july2013_QA** for any additional files and/or downloads associated with this article. You can also discuss this topic at **http://forum.nutsvolts.com**.

Radio Modification

Q Can you please help with a simple diagram for the FM radio shown in **Figure 1**? I would like to add an LM386 amplifier — less than or equal to one watt — with minimal parts and an eight ohm speaker to it. I'd like to eliminate the need for a 'crystal earphone,' using just one power supply — that of the FM radio or from a wall wart (9 VDC @ 100 mA).

Since I couldn't locate an original 365 pF variable capacitor at an affordable price, I substituted one rated 0-532 pF. Consequently, do I need to change the value of C4? Also, I couldn't find the two metal PNP transistors (25A220). Can you please recommend other substitutes, without modifying the entire diagram?

— Don Franklin



Q1 & Q2	PNP Transistor 25A220
R1	220 K OHM Resistor
R2	1.8K OHM Resistor
R3	82K OHM Resistor
R4	5K OHM Potentiometer
C1,C4,C6	50 P. F. Disc Ceramic Capacitor
C2	6.8 P. F. Disc Ceramic Capacitor
C3	365 P. F. Variable Capacitor
C5	5MFD 16V Electrolytic Capacitor
C7	25MFD 25V Electrolytic Capacitor
C8	.047 MFD Mylar Capacitor
C9	.02 MFD Disc Ceramic Capacitor
RFC 1	10 MHY R. F. Choke
T1	Audio Transformer 2K Center Tap 10K OHM

■ FIGURE 1.

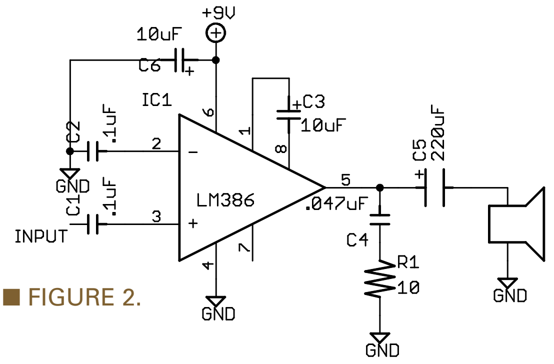
A This is not the usual FM radio; it is an AM radio that is able to receive FM by "slope detection." When the radio is tuned off center frequency, the slope of the response curve converts the FM to AM, which is then detected.

The LM386 datasheet has several circuits that you could use. The simplest (minimum parts) has a gain of 20 which is not enough to drive the speaker. The circuit with a gain of 200 is the one to use. In **Figure 2**, I show how it should be used in this case.

C4 — called a padder — is used to reduce the capacitance of C3 and thereby reduce the tuning range. It is not necessary to change C4, but if you want the same tuning range, the capacitance of series capacitors is calculated similar to parallel resistors:

C=C3*C4/(C3 + C4) = 365pF*50pF /(365 + 50) = 44pF

Here's a little algebra to find the value of C4 with 532 pF:



■ FIGURE 2.

C4=C*C3/(C3-C) = 44pF*532pF /(532-44) = 48pF

47 pF is an available value and 50 pF is difficult to find, anyway.

I recommend that you reverse the polarity of C5, C7, and the battery, and use an NPN transistor like PN222 or 2N3904

Beginner's Question

Q What would you recommend to start learning if you are completely new to electronics and want to start?

— Andrew Reardon

A You should know algebra to be able to understand explanations and calculate values of components. Depending on how far you want to go, you could continue on with calculus. You should study basic electricity, DC and AC circuits, impedance, reactance, resistance, and that stuff.

You will want to know about semiconductors, diodes, transistors, triacs, sidacs, op-amps, and logic. If your interest is computers, Boolean algebra will be useful and you will want to become familiar with the logic elements: AND, NAND, OR, NOR, XOR, etc.

If your interest is programming, you will eventually need to know all this in order to make the machine work in an analog world.

Start small and build big. I spent 10 years in school and 30 years in industry, learning new things all the time. My first project at 13 or 14 was a crystal radio; it worked and I was hooked.

Jacob's Ladder Circuit

Q Thanks for your help on the high voltage generator (June 2013 issue), and the kind words about my oscillator circuit. I try to research answers and solutions for problems to teach myself, and every once in a while I come up with something that works well.

I moved R2 from 12V and connected it to the Q2 emitter, omitted D4, and the snubber R3 and C4 (*refer to Figure 7 in the June 2013 issue*).

Out of curiosity, I connected a neon bulb and a 100K resistor across the power supply leads to the coil to see how severe back voltage spikes were.

As the spark started, the neon light lit and Q2 failed. After checking what replacements were on hand, I discovered there was a NTE2394 N-channel enhancement mode MOSFET in my stock pile (450V 12A). So, I decided to try it.

MOSFETs are not very familiar to me; I have not used them much. However, at first glance, I think it will so a better job. I am also considering Schottky diodes, but 300-400 volts seems to be hard to find.

Is there any advantage in using Q1 (the 2N3904)? If so, where should the collector be tied? Or, is my first circuit the better one? Either way, are my resistor values for R1, R2, and R3 okay?

— Mike

A I have suggested some changes to your circuit that are shown in **Figure 3**. You don't need D3 because there is a parasitic diode in the MOSFET and if the circuit is working correctly, it does nothing anyway.

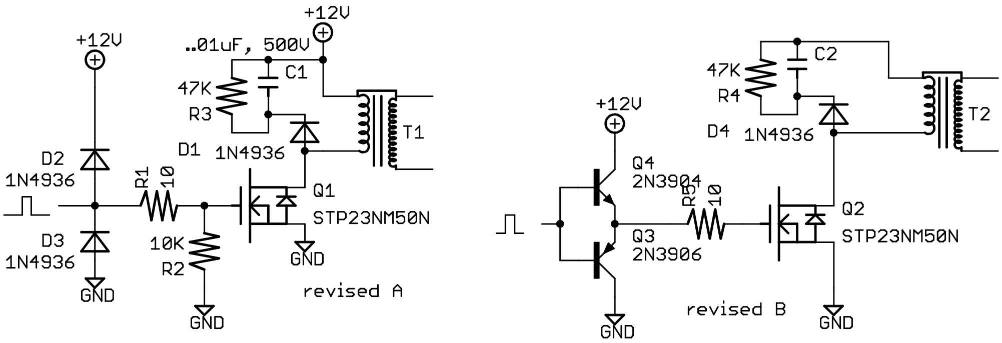
D1 in my schematic (**Figure 3A**) charges C1 to the peak of the flyback, so you can read the peak voltage with a DC meter. If you start with a narrow pulse, the peak voltage will rise as the pulse width is increased. The voltage will stop rising when either the current is limited by the resistance of the coil or the core is saturated. In either case, further increase in pulse width just wastes power.

You could drive the gate directly from the logic (no resistors or diodes) because the logic has limited drive capability. If you had a driver capable of amperes, then a series five or 10 ohm resistor would be needed to limit the current. A MOSFET has a DV/DT limit, above which it will break down.

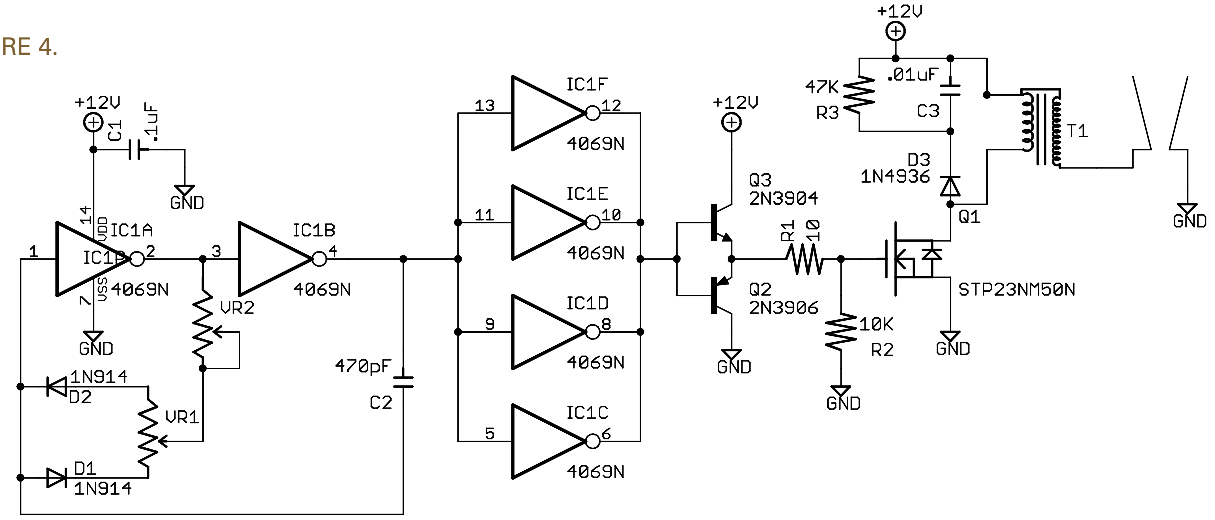
In my circuit in **Figure 3A**, the diodes (D2, D3) and resistor (R1) are to protect the driver in case of a drain-gate short. R1 needs to be small because you want to discharge the gate-source capacitance quickly in order to turn the MOSFET off quickly.

In **Figure 3B**, the NPN-PNP driver provides more current to the MOSFET turning it on and off faster, and thereby reducing power loss. The series 10 ohm resistor limits the current to one amp, although the transistors probably can't

■ FIGURE 3.



■ FIGURE 4.



produce that much current.

In your test with the neon bulb and 100K resistor, there was not enough load on the secondary to limit the voltage. In normal operation with a spark plug load, the voltage will be about 300 volts; with no load, the voltage could be 600 or more on the primary side. I agree that a MOSFET is the better choice, and high voltage units are available. I have not seen a Schottky diode rated over 100 volts, so if you need a high speed, high voltage diode, I suggest something like a 1N4937.

I have reproduced your revised schematic in **Figure 4**; it looks good to me. Let me know how it performs.

Electret Microphones

Q Recently, you clearly indicated what type of microphone I should use for my one watt amplifier that is mounted at the focal point of my parabolic ("salad bowl") reflector. This time, I have two different amplifiers I wanted to connect to ceramic/crystal mics, but they're rather expensive (on eBay) when available — I assume that the next best thing would be an omni-directional electret condenser microphone. Can you please specify which ones would best fit the two amplifiers, and include on the diagram how to connect them?

— Nate

A When reading the microphone specs, the lower dB numbers are the higher output. The dB is relative to one volt, so -36 dB is equivalent to 16 millivolts. Three or four volts is needed to drive a speaker, which indicates a gain of 4/.016 = 250. A preamp with a gain of 10 will provide more than enough gain, along with an amp with a gain of 200. If you keep in mind that -6 dB is half the voltage, then it is easy to calculate dBs: zero dB = 1V; -6 dB = .5V; -18 dB = .125V, etc.

The electret microphone has a permanently charged dielectric, so it does not need any applied voltage. These microphones from Mouser do require an applied voltage to operate the internal amplifier, however.

The 2.2K output impedance is due to the amplifier circuit. The maximum voltage is usually 10 volts so you could connect it through a load resistor to the nine volt battery, except that the amplifier would no doubt oscillate due to feedback through the power supply.

I recommend a 6.2 volt zener and 10 µF bypass for isolation (see **Figure 5**). Scrap those circuits you were contemplating and use the LM386 circuit.

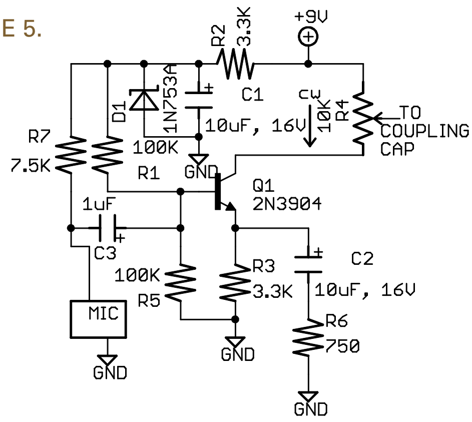
A Stable, Accurate Oscillator

Q I did the math for 49.848230 kHz for a signal generator and had a heart to heart talk with one of the Circuit Specialist engineers who told me that to get this kind of resolution, I would need a DDS type of generator. I saw the Velleman 1 MHz pocket function generator. It does have a DAC resolution of 10 bits and a frequency range of 1 Hz to 1 MHz ± 0.01%; model #HPG1. All Spectrum Electronics has it for $129.99. This is a direct digital synthesis generator. What do you think?

A I looked up the Velleman model HPG1 signal generator and the accuracy of .01% will still leave a 4.984 kHz error in frequency. You could perhaps adjust the frequency to 49.848230 kHz, but you will need a frequency counter with an eight digit display and .00001 % accuracy. Even with such a counter, it is not likely that the Velleman signal generator will have the stability to stay on frequency.

What you need is a quartz crystal oscillator, but I don't think you'll be able to build it and get it on frequency. So, I suggest that you purchase a voltage tunable (VCXO)

■ FIGURE 5.



oscillator. For the Crystek VCXO, frequencies start at 1 MHz so you will need to specify a frequency of 1595143360 Hz and divide by 32 to get your desired frequency. The manufacturer will tell you that they can't guarantee frequency to that accuracy, but they should be able to get within 100 PPM so you can tune it. The Crystek part number is CVXO-018TX-50-FREQUENCY in MHz.

If you need a sine wave, I can design a filter for you. **NV**

NEW PRODUCTS

- HARDWARE
- SOFTWARE
- GADGETS
- TOOLS

FLEXIBLE RESOLUTION OSCILLOSCOPES

Pico Technology is now utilizing reconfigurable ADC technology to offer a choice of resolutions from eight to 16 bits in a single product.

Most digital oscilloscopes gain their high sampling rates by interleaving multiple eight-bit ADCs. Despite careful design, the interleaving process introduces errors that can make the dynamic performance worse than the performance of the individual ADC cores.

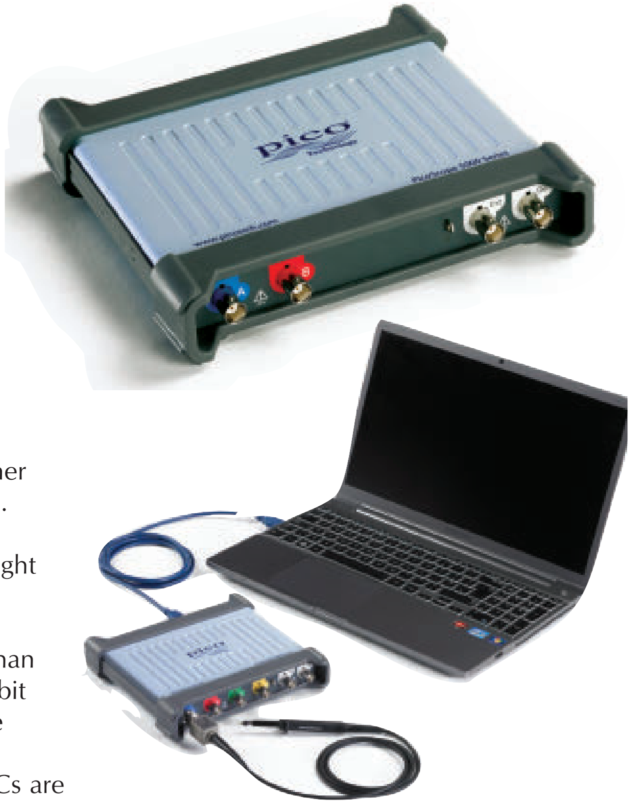
The new PicoScope 5000 Series scopes have a significantly different architecture in which multiple high resolution ADCs can be applied to the input channels in different series and parallel combinations to boost either the sampling rate or the resolution.

In series mode, the ADCs are interleaved to provide 1 GS/s at eight bits. Interleaving reduces the performance of the ADCs, but the result (60 dB SFDR) is still better than oscilloscopes that interleave eight-bit ADCs. This mode can also provide 500 MS/s at 12-bit resolution.

In parallel mode, multiple ADCs are sampled in phase on each channel to increase the resolution and dynamic performance. Resolution is increased to 14 bits at 125 MS/s per channel (70 dB SFDR). If only two channels are required, then resolution can be increased to 15 bits, and in single channel mode all the ADCs are combined to give a 16-bit mode at 62.5 MS/s.

Careful attention was required to support the high resolution modes (with low noise, low distortion, and bandwidth flatness), while maintaining the bandwidth, slew rate, and pulse response necessary for the faster eight-bit mode.

As well as flexible resolution, these oscilloscopes have ultra-deep memory buffers of up to 512 MS to allow long captures at high sampling rates. Also included as standard are advanced software features such as serial decoding, mask limit testing, and segmented memory.



The PicoScope flexible resolution oscilloscopes are priced from US$1,153 for the two-channel 60 MHz model with built-in function generator to US$2,803 for a four-channel 200 MHz model with built-in AWG. Prices include a set of matched probes, all necessary software, and a five year warranty.

For more information, contact:
Pico Technology
Web: **www.picotech.com**

POWER SUPPLY WITH MULTIPLE VOLT/AMP COMBOS

B&K Precision has expanded its DC system power supply offerings with the introduction of their Model 9115 multi-range programmable DC power source. The 9115 can deliver up to 1,200W in multiple combinations of voltage up to 80V and current up to 60A. With a high density, compact 1U form factor and standard 19" rack-mountable design, this DC power supply can replace multiple power supplies on a bench or in a rack, making it suitable for a variety of benchtop or ATE system applications.

The 9115's multi-ranging capability automatically recalculates voltage and current limits for each setting, and can provide wide voltage and current operating ranges without forfeiting high output power, thus saving valuable bench space by eliminating the need for large bulky power supplies or multiple units.

On the front panel, the 9115 features a high resolution, 1 mV/1 mA VFD display, independent voltage and current control knobs, cursors, and a numerical keypad for

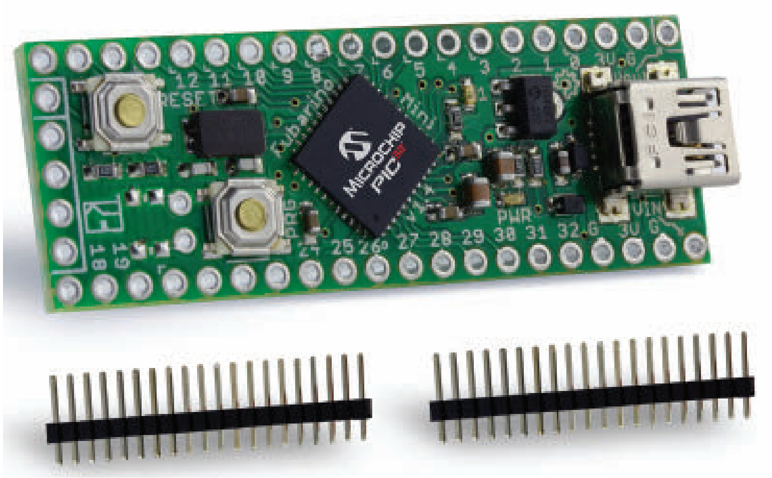
direct data entry. The instrument also provides internal memory storage to save and recall up to 100 different instrument settings.

For programming and remote control, the 9115 offers a plethora of features to the user such as: adjustable voltage rise and fall times, sequence (list mode) programming, master/slave mode for series or parallel connectivity, remote sense, a DB25 analog interface with control and monitoring functions, and standard USBTMC-compliant USB, RS-232, GPIB, and RS-485 interfaces supporting SCPI commands.

The 9115 also provides configurable overvoltage and overpower protection limits, and a front panel key-lock function to prevent accidental damage to a device under test. The 9115 multi-range programmable DC power supply is listed at $1,695.

For more information, contact:
B&K Precision
Web: **www.bkprecision.com**

ARDUINO COMPATIBLE chipKIT BOARDS



Microchip Technology, Inc., has announced the expansion of its Arduino™ compatible chipKIT™ platform ecosystem, with new tools from partners Digilent, Inc., the Fair Use Building and Research (FUBAR) Labs, and Schmalz Haus LLC. These new tools are based on the 32-bit PIC32 microcontrollers in prototyping-friendly, low pin count SOIC or SPDIP packages — which were previously more common in eight-bit microcontrollers — for the Arduino community.

This enables users — including hobbyists, students, and professionals — to benefit from the PIC32's high performance, memory, and integrated peripherals while using the basic hobbyist prototyping equipment that is found in most home workshops.

Digilent's chipKIT DP32 board features basic I/O and interface components, expanding the 32-bit chipKIT ecosystem while providing a low cost, Arduino-compatible development platform with a great out-of-the-box user experience.

The chipKIT Fubarino™ Mini board (from the partnership between FUBAR Labs and Schmalz Haus) also provides a great option for Arduino-compatible development with 32-bit PIC32s, using a prototyping-friendly board form factor.

For users who wish to build applications without a development board, Microchip is also making the prototyping-friendly PIC32s in PDIP packages available with the preprogrammed chipKIT USB bootloader. Specifically, the preprogrammed PIC32MX250F128B is available in a 28-pin package. The advanced features on this 32-bit microcontroller include 40 MHz performance, 128 KB Flash, and 32 KB RAM, along with integrated peripherals for touch sensing, graphics, audio processing, USB, and advanced control applications.

The Fubarino Mini is available for $19.95 each. Digilent's chipKIT DP32 board is priced at $23.99.

The 32-bit PIC32MX250F128B preprogrammed with the chipKIT USB bootloader is available starting at $5.95 each, in single-unit quantities.

For more information, contact:
Microchip Technology
Web: **www.microchip.com**

If you have a new product that you would like us to run in our New Products section, please email a short description (300-500 words) and a photo of your product to:

newproducts @nutsvolts.com

GEIGER COUNTER WITH WAND

The GCA-06W and GCA-06 digital Geiger counters now available from Images SI, Inc., are accurate enough to be NRC calibrated and certified.

Priced at $479.95, the unit is useful for detecting and measuring radioactivity. The GCA-06W uses an external wand that houses the Geiger Mueller (GM) tube. The GCA-06 has an internal GM tube.

Some applications include:

- Education; classroom demonstrations and experiments.
- Emergency services and domestic preparedness, HAZMAT and compliance verification.
- Radiation screening and EMTs.

The Geiger counter detects the following types of radiation: alpha above 3 MeV; beta above 50 keV; X-rays above 50 keV; and gamma above 7 keV. The GM tube detector is Ne + halogen filled.

The GM tube has a .38 effective diameter 1.5-2.0 mg/cm2 mica end window. The LCD is 16 characters by two lines that provide an easy to read output. Switch selection allows the Geiger counter to measure and convert radiation (counts per second) into mR/hr or mSv/hr. Factory calibration insures accurate measurement.

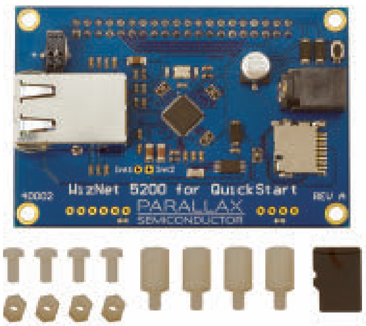
Secondary indicators for audio (clicks) and visual (LED) are included. There is also a headphone jack and a power jack for external power. The GCA-06W serially outputs the counts per second via a USB cable to Windows PC program for charting and recording measured radiation over time. A USB-TTL serial cable is included.

Some additional features are:

- Resolution and range: one count per minute (CPM); 5,000 counts per second (CPS).
- Radiation resolution and range .001 mR/hr - 1,000 mR/hr; metric scale .01 uSv/hr - 10.0 mSv/hr.
- Low battery indicator lights when battery voltage drops below seven volts.

For more information, contact:
Images SI, Inc.
Web: **www.imagesco.com**

WIZnet ETHERNET BOARD



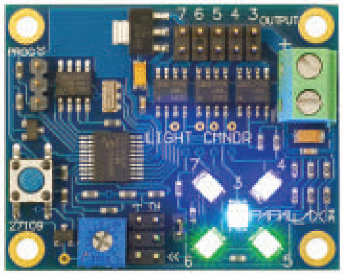
The WIZnet W5200 for QuickStart, now available from Parallax, allows your QuickStart board to serve data to an Ethernet network or even the Internet. The WIZnet W5200 chip provides an Ethernet to SPI bridge which is interfaced with Parallax's eight-core microcontroller, the Propeller P8X32A. The data served to a network can range from web pages, serial data, email, and more.

There are several demo programs available for download to get you started, including a webserver capable of serving dynamic data from the SD card to your favorite browser. Advanced users can develop firmware that allows the W5200 board plus QuickStart to respond to or generate any type of network traffic.

It's priced at $49.99.

LIGHT COMMANDER

The Light Commander — also available from Parallax — is a multi-output user programmable power controller. Originally designed for use as a light controller on the Parallax ELEV-8 quadcopter, it is equally well suited for any application requiring the



switching of high current under programmable control. Based on the proven technology of their BASIC Stamp 1 microcontroller, this product is both simple to use and versatile.

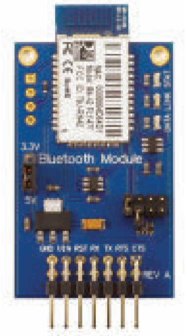
It is priced ad $29.99.

REED SWITCH KIT

Parallax's Glass Reed Switch Kit comes with everything necessary to give your microcontroller the ability to detect the presence of magnetic fields. The reed switch sensor itself is a low tech and time tested mechanism that makes use of two magnetically actuated reeds.

The retail price is $7.99.

BLUETOOTH MODULE

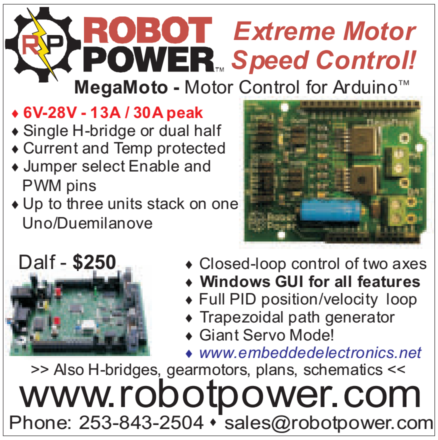


Finally from Parallax is the RN-42 Bluetooth Module which provides a low cost method for creating a wireless serial communication interface between two devices such as a microcontroller, PC, cell phone, or another module. The RN-42 can pair up with devices supporting Bluetooth SPP (Serial Port Profile) to establish a serial interface. The RN-42 is breadboard-friendly and is compatible with all 5V and 3.3V microcontroller platforms.

It is priced at $49.99.

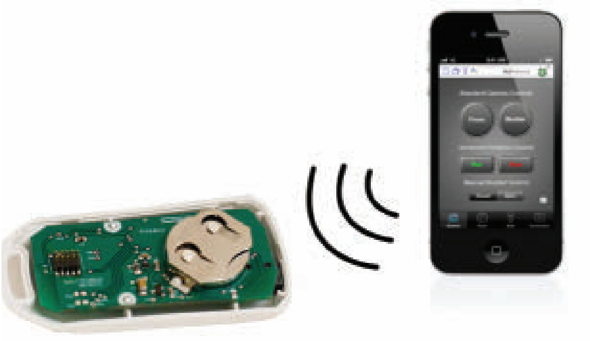
For more information, contact:
Parallax
Web: **www.parallax.com**

SHOWCASE



Programming the Raspberry Pi
Getting Started with Python
Simon Monk
Raspberry Pi User Guide
BUILD YOUR OWN FREE-TO-AIR (FTA) SATELLITE TV SYSTEM
DENNIS C. BREWER

BUILD A BLUETOOTH CONTROLLER FOR YOUR DIGITAL CAMERA



Self-portraits, group shots, time-lapse, and HDR made easy with a little help from your favorite iOS device.

By Michael Wieckowski, Ph.D.
info@shutterbugremote.com

Go to **www.nutsvolts.com/index.php?/magazine/article/july2013_Wieckowski** for any additional files and/or downloads associated with this article. You can also discuss this topic at **http://forum.nutsvolts.com**.

Sick of watching the blinking timer on your camera as you run across the room to get into a group portrait? Tired of holding the shutter down for a few minutes doing a long night exposure? Sure, you could buy a remote shutter release, but then you'd have another piece of equipment to learn how to use, carry around, and keep charged. In this project, we'll build a simple Bluetooth dongle using the latest low power technologies to let you control your camera directly from your iPhone or iPad.

Bluetooth 4.0 Is Unlike Any Bluetooth You've Seen Before

We're all familiar with Bluetooth. Your computer mouse has it, your phone has it, even your car might have it. In 2010, the latest update to the Bluetooth specification was introduced — referred to as Bluetooth 4.0. Along with legacy modes and high speed options, a new low energy provision was added. This specification — also known as BLE or Bluetooth SMART — is specifically focused on extremely low power consumption, reduced size and cost, and broad compatibility. You might see the logo for it on your new laptop or tablet (refer to **Figure 1**).

■ FIGURE 1.

A range of 50 meters is complemented by a data rate in the hundreds of Kbits/s and a battery life of months from a standard coin cell. BLE provides a means to link your smart devices to the real world through inexpensive, highly capable sensors. Just take a look at the fitness wrist bands, GPS trackers, and heart rate monitors that are already on the market using this technology.

In this article, we will focus on using BLE to create a wireless link between your smartphone and actually any project you can dream up. Specifically, we'll design a BLE circuit for controlling a digital camera remotely. Expanding or modifying the approach to work on a robot, garage door, etc., would be a straightforward task and a fun project for an advanced experimenter.

The Perfect Link To Control Your Digital SLR

For over a hundred years, the ability to remotely control a film camera's shutter using a cable has been a staple in professional photography. Such a mechanism enables the photographer to create long timed exposures, and to execute them without physically shaking the camera. So, how does one "reinvent the mousetrap" with Bluetooth? Replace the complex remote control unit with a smartphone, of course! By leveraging your phone as the user interface, you can plug an inexpensive and lightweight Bluetooth module into your camera to create a full featured wireless controller.

Choosing A Bluetooth Module

There are a variety of Bluetooth Smart modules on the market today, and even more on the horizon. Let's look at the wish list for our module:

1. Software stack built in, with an elegant interface.
2. Integrated antenna and front-end matching circuitry.
3. Certified by the FCC, Industry Canada, and any other regulating agencies.

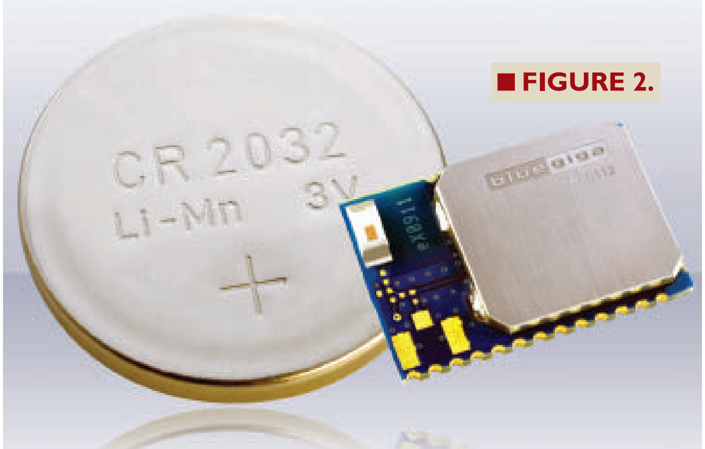
We will be utilizing a product from BlueGiga called the BLE112A (see **Figure 2**). The module itself is tiny — 18 x 12 mm — with surface-mount pads surrounding it. A chip antenna is built in, offering a range somewhere between 200-300 feet. It is certified by all the right agencies and can be added to a product without any further testing. Most importantly, an 8051 processor is already onboard running the Bluetooth stack.

Using a propriety scripting language called "BGScript," we can write our application firmware without the use of an additional microcontroller. All we'll need is a programmer from Texas Instruments (TI) to connect and upload our scripts. At under $20, it's an impressive set of features that will enable most hobbyists to quickly add Bluetooth Smart to all of their projects.

Some Bells And Whistles

To make our camera controller able to compete with the big guys, we're going to need a few more features for accurate timekeeping. For example, we may want to take a photo every hour for 10 days. Or, we might like to take a single photo 30 seconds from now. To enable this kind of flexibility, relying on the timers inside of the BLE112's processor is not the best solution. Instead, we'll use a real time clock chip (RTC) designed to do nothing but keep accurate time using the fewest number of electrons possible, and a precision crystal for reference. My RTC of choice is Microchip's MCP7952x. At only 700 nA from 1.8V, we'll be able to accurately time all of those delays and shutter exposures using a simple SPI communication bus.

Another important consideration is power management. The BLE112 automatically moves in and out of various modes to minimize current draw, and features a special output pin for pairing with a DC-DC converter. The datasheet presents a circuit from TI (keep in mind that there is a TI processor within the BLE112) for connecting this pin to a TPS62730 — a DC-DC converter optimized for running on a coin cell battery. In our design, we'll use this exact circuit to efficiently switch a lithium coin cell from 3.6V down to 2.1V, giving us a boost in battery life of about 10%.



■ FIGURE 2.

A Quick Intro To HDR

The human eye can see an exposure range of about 10 f-stops normally, and up to 24 f-stops given time to "adjust" to the scene. The ADC in your digital camera is typically 10 to 12 bits and when noise is considered, it can only capture a range of about five to nine f-stops. HDR (High Dynamic Range) photography is a technique to enhance the range of your camera by combining several images of different exposures.

A common approach is to capture three or more images — some underexposed by a few stops, and some overexposed by a few stops. Using a remote trigger is important since the camera should be mounted on a tripod and all of the images should be identical.

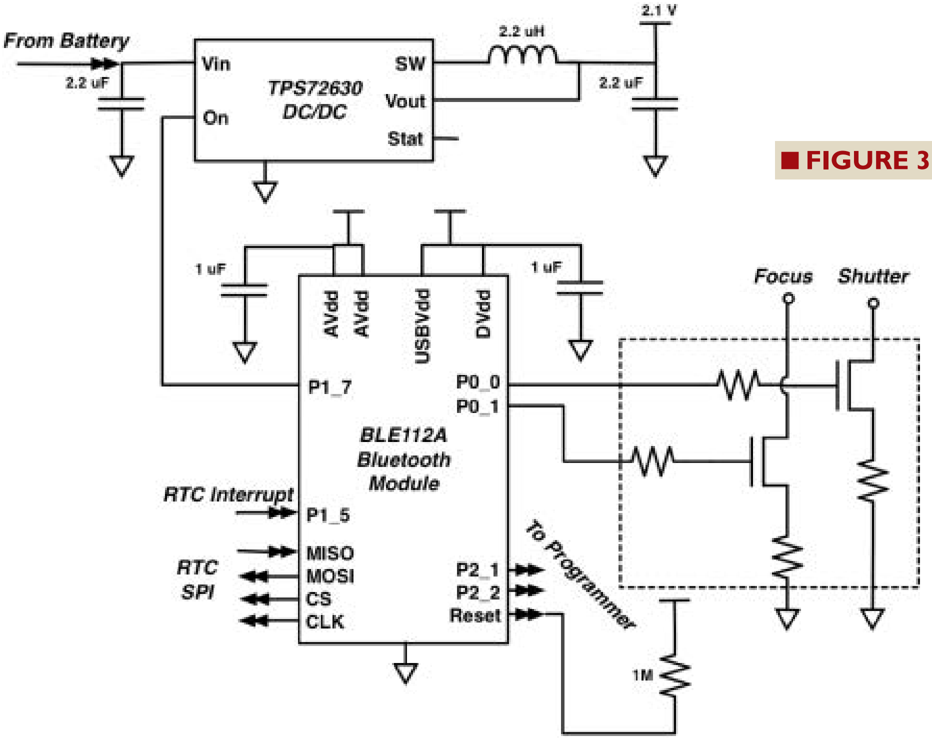
A post-processing program like Photomatix or Photoshop can then be used to combine these images and map them into a 16-bit format suitable for display on your screen. The result is a photograph with incredible depth and detail. Check out **stuckincustoms.com** for some great info.

The Circuits

Take a look at **Figure 3**. The BLE112A requires surprisingly few components to get up and running. To generate the 2.1V supply, the TPS72630 DC-DC converter is used along with a 2.2 µH inductor, an input capacitor, and an output capacitor. P1_7 on the BLE112A is the special pin used to enable the DC-DC converter, allowing its processor to save energy by requesting a regulated output voltage only when needed. P2_1 and P2_2 are used in conjunction with the RESET pin to interface to the programmer. TIs' "CC Debugger" is $49, and allows you to program the BLE112A from a computer using USB and a HEX file.

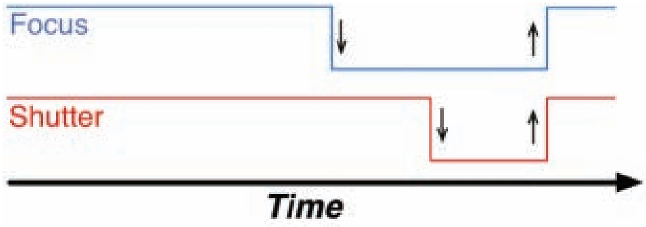
Most digital SLR cameras offer a shutter release port with two pins that mirror the action of pressing the shutter button on the camera. One pin corresponds to "focus" and the other to "shutter." If you were to press the shutter button halfway down, this would be the same as pulling the focus pin low and would focus the lens. Pressing it all the way down would first pull the focus pin low, then the shutter pin low, and then would take a picture.

We need a mechanism to separately pull these two



■ FIGURE 3.

■ FIGURE 4.



The Firmware

The complete firmware for the camera controller is over 1,000 lines of code. We'll go over the basic structure here and focus on a few key elements that will help you get a Bluetooth project up and running in no time.

Listing 1 shows the XML file to describe the hardware configuration. Lines 2 and 3 enable the BLE to go to sleep when needed. Line 4 enables a pullup resistor on P1_5 which is the interrupt line coming from the RTC. Line 9 instructs the BLE to use the DC-DC control pin. Line 10 sets up the SPI port to connect the RTC. We chose a 2 Mbps baud rate on channel 0.

Another required file is the GATT.xml. GATT is a type of Bluetooth profile for exchanging "Generic ATTributes." It allows us to define "characteristics" which are the basic building block of Bluetooth data transfers. A reduced version of the profile is shown in **Listing 2**. Here, we define a service called "cameraRemoteShutter" and we advertise it to the world. Our iOS app will be looking for this service to connect. Within the service, we create a characteristic called "shutterOpenClose." It is a hexadecimal number that is four bytes long (32 bits) and can be read from or written to. Our iOS app will write a one to this characteristic to open the shutter and a zero to close it. Our full GATT profile defines 16 distinct characteristics to handle such things as number of exposures, time between exposures, etc.

Listing 1 - Hardware.xml

```
1.<hardware>
2.    <sleeposc enable="true" ppm="30" />
3.    <sleep enable="true" />
4.    <port index="1" tristatemask="0x80"
      pull="up" />
5.    <usb enable="false" />
6.    <txpower power="15" bias="5" />
7.    <script enable="true" />
8.    <slow_clock enable="true" />
9.    <pmux regulator_pin="7" />
10.   <usart channel="0" mode="spi_master"
      baud="2000000" polarity="negative"
      phase="0" endianness="msb" endpoint="none" />
11.</hardware>
```

pins to ground on command as shown in **Figure 4**. Back in **Figure 3**, two biased NPN transistors are used as pulldowns with their bases connected to P0_0 and P1_1 of the BLE112A. By setting these two pins high or low, we can control the shutter and focus. Instead of two discrete transistors, NXP offers a device called the PUMH10 which contains everything in a single package.

The last connection required in our system is the communication bus to the RTC. The RTC will act as an SPI slave, and P1_5 will be used as an interrupt allowing the MFP signal to alert the BLE112A that a timer has expired. The RTC itself only needs power and a crystal as shown in **Figure 5**.

Since our project is only useful when connected to a camera, it makes sense to turn it on and off using the plug itself. To do that, a 2.5 mm audio connector with a detection contact is used. Looking at **Figure 6**, when the plug is *not* inserted, the detect pin is shorted to the shutter pin. Since the shutter line has a 1M pulldown resistor, the input to the inverter will be zero resulting in a battery level output. This will keep the PFET off and shut down the DC/DC converter. When the plug *is* inserted, the detect pin will float high resulting in a zero on the PFET gate. This will connect the battery to the DC/DC converter and turn on the power.

FIGURE 5.

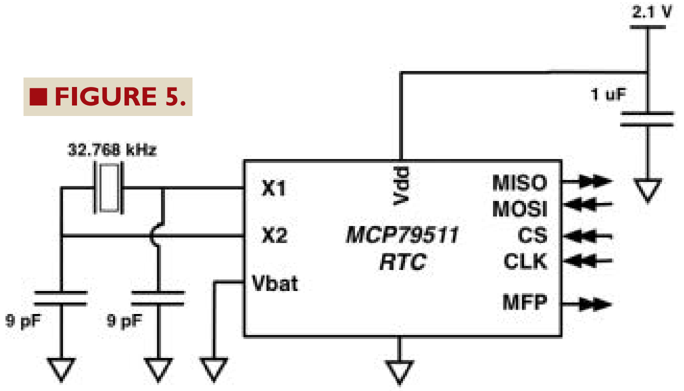
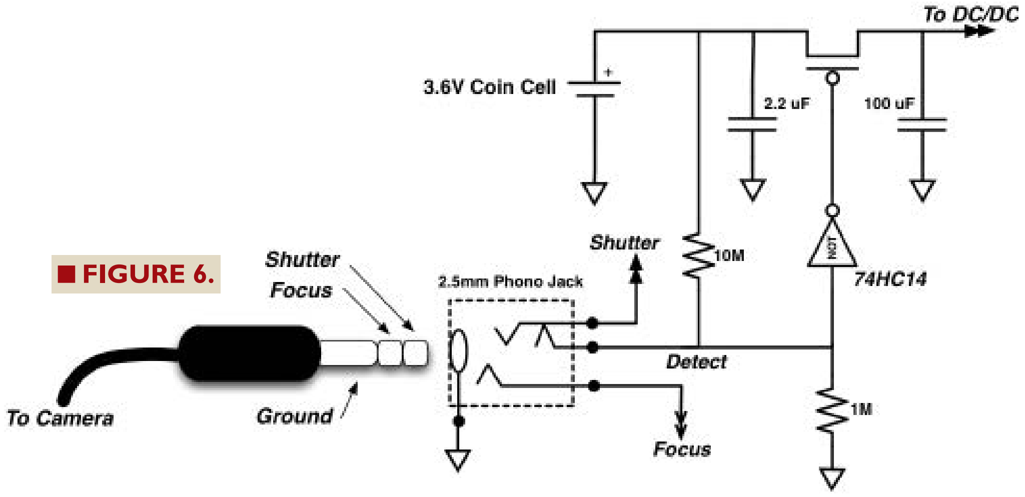


FIGURE 6.



You may have noticed that every element of the GATT profile has a UUID associated with it. This is just a long hexadecimal string required to make each service and characteristic unique. There are online UUID generators that you can use to create them.

Listing 2 - GATT.xml

```
<configuration>
   <service type="primary" uuid=
     "98BF9A5A0DD011E2A376CC4A6188709B" id="
     cameraRemoteShutter" advertise="true">
   <description> Shutter Remote Control Service
     </description>
   <characteristic uuid="198502EEA8E3
     4D16B79C89928367BEDF" id="
     shutterOpenClose">
           <description> Shutter OpenClose
             </description>
           <properties write="true" read="true"/>
           <value length="4" type="hex"> 0
             </value>
   </characteristic>
</configuration>
```

The last step is to write our firmware using BGScript. This scripting language is event based; sections of code are executed in response to various Bluetooth events defined in the BGScript programming guide. **Listing 3** shows the first event that we need to handle which is called “system_boot.” This event will execute when power is applied to our circuit, and is the place to do general setup tasks. Lines 3 through 10 deal with setting up the direction (input or output) of various pins on the module, as well as their levels (high or low). Line 11 tells the module that we want to have an interrupt (IRQ) generated when bit 5 of port 1 is pulled low by the RTC. Lines 13 through 15 make our first transaction on the SPI bus.

First, CS is pulled low to select the RTC as the slave we are talking to. Then, we do a three byte SPI transfer to start the RTC crystal oscillator. The last few lines set up the Bluetooth advertisement interval and connection modes.

The next event that we need to handle is the most interesting; it’s called “attributes_value.” This event is the key to exchanging data between the BlueGiga module and an iOS application. Back in **Listing 2**, we created a GATT characteristic and gave it a meaningful name of

Listing 3 - BGScript event called when the system powers up.

```
event system_boot(major, minor, patch, build, ll_version, protocol_version, hw )
    #set up the ports
    call hardware_io_port_config_direction(0, $3B)

    #cs needs to start out low, and then go high to enable the RTC
    call hardware_io_port_write(0, $3B, $0)   #all low
    call hardware_io_port_write(0, $3B, $10)  #only CS is high

    call hardware_io_port_config_direction(1, $83)
    call hardware_io_port_write(1, $83, $80)   #set 1_7 high for DCDC on
    call hardware_io_port_config_irq(1, $20, 1)
    #start the RTC by setting ST flag
    call hardware_io_port_write(0, $10, $00)   #CS low
    call hardware_spi_transfer(0,3,"\x12\x01\x80") #RTC running
    call hardware_io_port_write(0, $10, $10)   #CS high
    call gap_set_mode(2,2)
    call gap_set_adv_parameters(1600, 1600, 7)
    call sm_set_bondable_mode(1)
end
```

"shutterOpenClose." When you run the BGScript compiler, it replaces all of these meaningful names with sequential numbers called "handles" that you can refer to in your scripts. In our specific example, shutterOpenClose is given a handle value of 17. In **Listing 4**, you can see how these handles are used.

When a characteristic is modified, both its handle and value are passed into this event. If our iOS app writes a value to shutterOpenClosed, this event will be called with a handle of 17. So, all we need to do is check the handle value with an IF statement (line 2) and then act accordingly.

You can see that in line 4 if the value written was 0, we close the shutter. If the value written was 1, we open the shutter. Voila! We now have a wireless link to our hardware.

The last event we need to look at is named "hardware_io_port_status." In **Listing 1**, we told the compiler that we wanted an IRQ whenever the RTC pulled P1_5 to ground. **Listing 5** shows this event, and provides us a way to react. Since we only have one IRQ, we can assume it was the RTC that created it. Lines 2-4 send a command to the RTC over the SPI bus that says "I got the IRQ; clear the flag." After this, we are free to do something useful like take a picture (open and close the shutter) and maybe send the RTC commands to initiate another timer.

Listing 4 - BGScript event called when a characteristic value is changed.

```
1.event attributes_value(connection, reason,
  handle, offset, value_len, value_data)
2.  if handle = 17 then
3.    valueWritten = value_data(0:1)
4.    if valueWritten = 0 then
5.      call hardware_io_port_write(0, 3, 0)
        #close the shutter
6.    else
7.      call hardware_io_port_write(0, 3, 3)
        #open the shutter
8.    end if
9.  end if
10.end
```

Listing 5 - BGScript event called when a timer IRQ is created by the RTC.

```
1.event hardware_io_port_status(delta, port,
  irq, state)
2.  call hardware_io_port_write(0, $10, $00)
    #CS low
3.  call hardware_spi_transfer(0,3,"\x12\x16\
    x70") #clear the interrupt flag
4.  call hardware_io_port_write(0, $10, $10)
    #CS high
5.  #do something useful here
6.end
```

Examples of Remote Shutter Triggers

Digital cameras can be triggered remotely in three different ways. The most common is using the universal shutter release port. There are a variety of physical connectors for different vendors and camera models, but the signals are universal. The second way is using USB.

Most modern cameras have some sort of digital port for transferring photos and controlling different functions. The problem here is that not only are the connectors all different, but the protocols are also proprietary. This means you'll need a trigger that is compatible with your specific camera.

The last way is using infrared. Many cameras have a small infrared sensor that can be used to trigger the shutter. These devices often suffer from issues with line of sight and interference on sunny days. Here are a few popular examples:

Wired: Canon RS60, Triggertrap Mobile,
Pixel Wired Release
Infrared: Canon RC-6, Photive ML-L3
RF Wireless: Yongnuo RF-603, Vello Freewave,
Pixel Pro
Bluetooth Smartphone: Shutterbug Remote, Satechi
Bluetooth Smart Trigger, Timelapse+

The Software

We've got hardware, we've got firmware, and now we need some iOS software. Why iOS you ask? As of right now, it is the only platform allowing access to Bluetooth LE through its API. It's likely that Android and others will catch on sometime this year, but if you want to get started today, iPhones and iPads are the only way to go. In addition, if you want to build a BTLE app and offer it in Apple's App Store, you won't need to go through any other certification such as "Made For iPhone" (MFI).

To get your phone talking to your hardware and begin testing, I recommend starting with one of the many BLE testing apps that are free from the app store. BLE Utility is one great example. These apps will let you scan for devices, read and write characteristics, and perform some other basic functions. To make your own app from scratch, you'll need a few things to get started: a good understanding of iOS programming; a $99 membership to the Apple developer program; and one of Apple's demos such as the Heart Rate Monitor App.

Putting It All Together

Even though the BLE112 is a surface-mount module, it's easy to solder a few wires to it and plug into a breadboard. With a 3.3V battery and a CC debugger, you can begin programming and testing your BLE module immediately. Once your prototype is working, I recommend skipping straight to a printed circuit board (PCB). The DC/DC converter and the RTC are quite small. By the time you buy the appropriate SMT adapters for prototyping, you'll quickly approach the cost of a barebones PCB.

For this camera project, I made a two-layer PCB using Eagle that is only two inches by one inch. The top layer hosts the DC/DC converter, CR2032 battery, and the debug header. The bottom layer holds the Bluetooth module, the RTC and crystal, and the 2.5 mm jack with power circuitry. (These PCB files are available at the article link.) You'll notice in these files that on the top and bottom layers, there is a cutout where the Bluetooth antenna sits to prevent any antenna degradation. The entire PCB fits into a plastic enclosure; I used the FB-45 from Polycase.

Conclusion

The size and range of these modules is truly impressive, and battery life is nothing short of amazing. I've been running one in a test for over three months and 20,000 photos, with no appreciable drop in battery voltage. In general, the Bluetooth LE protocol has far exceeded my expectations and I'm excited to see what clever implementations the hobbyist community is going to come up with. **NV**

PARTS LIST

MANUFACTURER (Mouser #)	DESCRIPTION
BLE112-A (603-BLE112-A)	BlueGiga BTLE Module
M50-3600542 (855-M50-3600542)	Harwin 10-pin Header
PUMH10 (771-PUMH10115)	Dual NPN Transistors
TPS72630 (595-TPS62730DRYT)	TI DC-DC Converter
LQM21PN2R2MC0D (81-LQM21PN2R2MC0D)	2.2 µH Inductor
771-BSS84-T/R (771-BSS84-T/R)	P-Channel MOSFET
MCP79511 (579-MCP79511-I/MS)	Real Time Clock
ABS10-32.768KHZ (815-ABS10-32.768K9T)	32.768 kHz Crystal
C1608C0G1H090C (810-C1608C0G1H090C)	9 pF Ceramic Capacitor
GRM188R60J225KE19J (81-GRM188R60J225KE9J)	2.2 µF Capacitor
GRM31CF50J107ZE01L (81-GRM31CF50J107ZE1L)	100 µF Ceramic Capacitor
C1608X7R1A105K (810-C1608X7R1A105K)	1 µF Ceramic Capacitor
301-10M-RC (301-10M-RC)	10M Thick Film Resistor
ERJ-3EKF1004V (667-ERJ-3EKF1004V)	1M Thick Film Resistor
STX-2550-5NTR (806-STX-2550-5NTR)	2.5 mm Phone Jack
SN74AHC1G14 (595-SN74AHC1G14DBVR)	Schmitt Trigger Inverter
CC-Debugger (595-CC-DEBUGGER)	Flash Programmer

Enclosure directly from **www.polycase.com**

FB-45-0	Polycase Plastic Enclosure
BATT-HDR-FB	Polycase 2032 Coin Cell Holder

Complete kits are available from the NV Webstore at ***http://store.nutsvolts.com****. Be sure and look for the custom iPhone app called Shutterbug Remote" at your favorite app store.*

BUILD AN EMERGENCY RADIATION MONITOR ALARM

By Ron Newton
sjnewt@att.net

Go to **www.nutsvolts.com/index.php?/magazine/article/july2013_Newton** for any additional files and/or downloads associated with this article. You can also discuss this topic at **http://forum.nutsvolts.com**.

When I wrote the article on a recording radiation counter back in the March 2013 issue, I had planned to follow up right away with an inexpensive version of a radiation monitor. I even had circuit boards made. However, upon testing, I found it did not work.

For two months averaging four hours a day, I toiled away, trying to get it to perform correctly but only came away frustrated. I tried circuit after circuit, including one that was for sale by a local kit maker. My wife calls me stubborn, but I prefer the word tenacious.

I breadboarded schematic after schematic, and put in SPICE models with no success. Several types of pin diodes were tried and tossed. My main problem was getting a gamma source for testing. Lantern mantles emit mostly alpha and beta rays with a few gammas. I finally managed through some college connections to obtain both a Cesium and a Cobalt source. In retrospect, it was the breadboarding and the absolute necessity of having the unit shielded which affected my progress.

This is an inexpensive surface-mount project that is good for beginners to start with. When designing a new project, I always start off with pre-specifications. For this project:

1. *I have a price point in mind.*
2. *The project should be self-calibrating, eliminating any potentiometers.*
3. *It must fit in an LED flashlight.*

Some Theory And Other Stuff

Gamma rays are high energy photons. The energy of the photon is measured in electron volts; keV or MeV. When using pin diodes to detect these photons, a major problem is that you must have enough energy to go through the substrate to cause the diode to conduct. This seems to be in the range of 60 keV or below, depending on the pin diode.

Pin diodes seem to be the reverse of intuitive thinking. For example, 10 keV is 100% effective. However, above 60 keV is just 1% effective. I'm figuring this is because the higher the keV, the shorter the wavelength so it squeezes through the substrate.

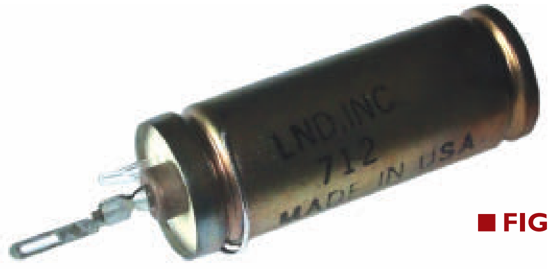
You may recall that my previous article was a full blown radiation counter for alpha, beta, and gamma rays, as well as any other high energy particles (e.g., X-ray, cosmic rays, etc.). For those who are interested in the difference between alpha, beta, and gamma rays, here is a brief description.

An alpha ray is made of two protons and two neutrons (which is the same as a helium molecule) and has mass. They can be blocked by a sheet of paper. The beta ray is made of high speed electrons or positrons, both having mass. They can be blocked by a sheet of aluminum. A gamma ray is a high energy photon with no mass (unless someone knows differently). They can be blocked by lead sheets.

With everything that's going on here at home and abroad, I feel the device presented here is a good addition to an emergency kit since we simply don't know what the future may bring, and whether or not it may become necessary to be able to detect radiation.

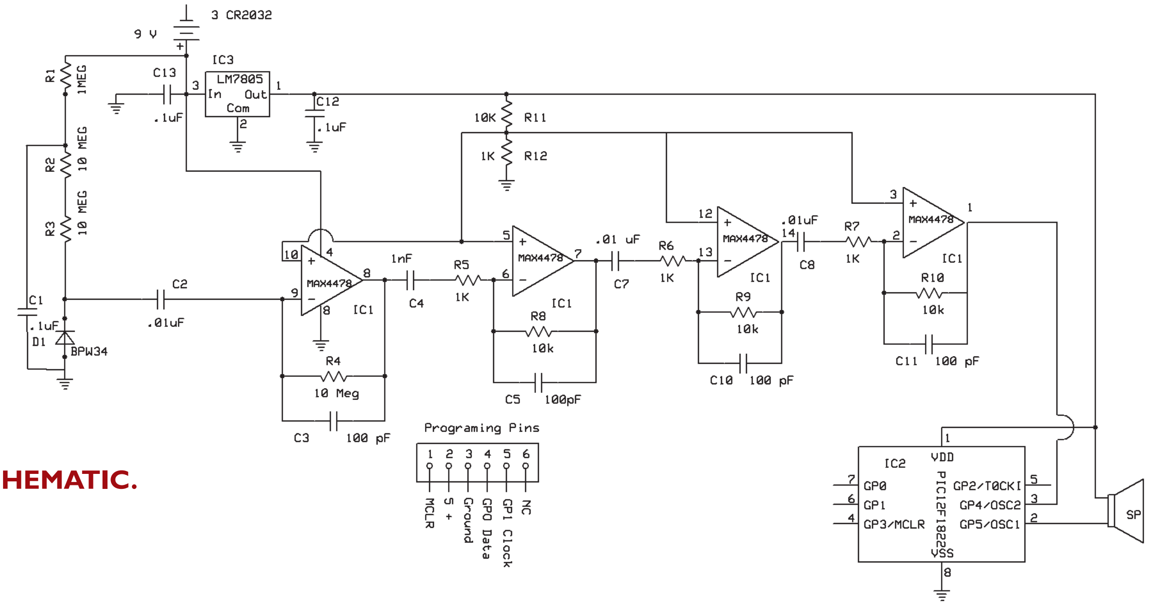
The batteries have a shelf life of over 10 years and are easy to find. This unit uses three CR 2032 coin batteries. Although there are several similar kits out there, this one has its own printed circuit board (PCB) which makes mounting the components easy. Plus, it comes in a pocket size shielded aluminum case measuring 3.5" in length and 1" in diameter. Its switch is a pushbutton on the bottom.

This project costs less than similar kits and needs no adjustment. It will click once each time it receives a high energy particle. If there is a sizable amount of radiation, it will sound the alarm.



■ **FIGURE 1.**

In my March 2013 article, I used a mica window Geiger tube (**Figure 1**) which was extremely sensitive; enough for detecting alpha and beta particles. (Geiger tubes require voltages above 300 volts to operate and the



■ **SCHEMATIC.**

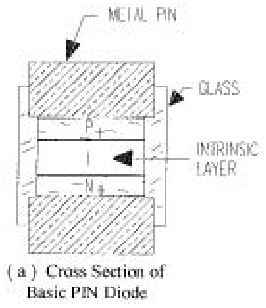


FIGURE 2.

tubes are expensive and fragile.)

Pin diodes are less than a dollar each, and recognize higher energy forms of radiation (e.g., gamma rays and X-rays), however they will not measure alpha or beta rays. This is due to the thickness of the silicon wafer, substrate, and window (refer to **Figure 2**). If medical isotopes (used in hospitals) happen to be present in an explosion, you most likely will be able to detect these as well, since they emit both beta and gamma rays. Their gamma output energy depends on the isotope used.

EMERGENCY RADIATION ALARM HINTS AND TIPS

To view and/or change the PCB files and schematic, go to **www.expresspcb.com** and download their free CAD software. There is no obligation and you will not be harassed by emails.

Microchip provides free software for programming their microprocessors at **www.microchip.com**.

SURFACE-MOUNT SOLDERING

Probably the most difficult part of surface-mount soldering is looking for the part after you squeeze it with the tweezers. It usually pops off and flies across the room. I keep extras around for this purpose. A couple of things can help, though. Put white butcher paper on your bench. Get a white apron (steal one from your wife) and sew two 1/2" pieces of Velcro™ on the corners and attach the hooks under the bench. Put the apron on and stick the corners to the bench. Use a magnifier or loupe for soldering and inspecting the parts.

Resistors are marked, however, most capacitors are not. So, make sure you only open what capacitors you are going to use. If I have a .1 µF strip, I solder all of them before going to the next group. Use a small dish when you peel the part into its center hole. The parts often slip when picking them up. This way, they fall into the dish. I use Chip Quik tack flux on the pads and curved tweezers to place the part on the solder pads. Normally, there is just enough solder on the trace to tack the part.

Once tacked, I solder the other side and then the tacked side. Use fine solder braid to remove excess solder. The flux can be removed with alcohol.

Harbor Freight used to carry refillable able spray paint cans that I fill with alcohol and compressed air. Make appropriate sized stiff brushes out of plumbing flux brushes by cutting off about half of the bristles.

PROGRAMMING

The square pin is pin 1 of the programmer. You can change the alarm rate by changing line 132 of the asm file.

MORE PROJECTS

Check out **www.newtsbooks.com** for more scientific projects.

Electronics

I used an LM7805 as the voltage regulator for two reasons. First, the microprocessor we're incorporating here (the PIC12F1822) uses five volts. Second, the circuitry is very sensitive to voltage changes since the PIC is also being used as a voltage comparator.

The BPW34 pin diode in this configuration is at high impedance when there is no light radiation. When light hits the diode, its impedance drops. When a gamma photon hits the pin diode, it gives a very small positive short pulse.

The MAX7748 has four high impedance, low capacitance amplifiers. The first amplifier boosts this pulse, which is fed into its inverting op-amp pin. This first op-amp amplifies the signal up to 6,000 times, depending on the frequency of the pulse. C2 changes is reactance with frequency.

Capacitive reactance $$X_c = \frac{1}{\omega C} = \frac{1}{2\pi f C}$$

e.g., a .0001 per second pulse (10,000 Hz) has a reactance of 1,592 ohms. At a one per second pulse (1 Hz), its reactance is 15,928,566 ohms. The rest of the op-amps also act as low pass filters and amplifiers. The MAX7748 circuit was taken from the manufacturer. They used a 4.7 pF capacitor for C3. I ended up using 100 pF due to excessive oscillation caused by feedback.

Each amplifier boosts the previous signal 10 times, depending on the frequency. The noise generated by the pin diode, R2-R3, and R4 is very high frequency and has to be filtered. The final output is fed directly into a PIC12F1822.

This PIC was chosen because it has an internal clock of 30 MHz which helps measure short pulses. It is configured to generate a longer pulse than what a gamma ray produces to get a good click. It also counts the number of clicks per minute (CPM) and sounds its alarm for five seconds if clicks are over 100 CPM. The average count of the pin diode is about 10 CPM.

Since we're using a newer PIC, it can't be programmed with a PICkit 2 or PICkit Starter; you'll need a PICkit 3 to program it, which is available in the *NV* webstore.

Instead of using a potentiometer and voltage comparator, I averaged 1,024 counts which set the baseline for noise, and then subtracted a known tripping voltage to the noise level. The A/D converter is set to a voltage reference of 1.024 volts for maximum sensitivity, and checks to see if a pulse is below a preset level. It provides a pulse stretcher for the speaker for volume.

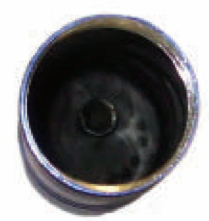
The CR2032 batteries have an output of 265 mAh. The unit draws about 1.2 milliamps. The unit should last about nine days continuously running, depending on the number of alarms.

Building The Chassis

The chassis is a Harbor Freight item # 69065 sells $3.49 for two. I have seen them as low as two for $1.99. This is a good enclosure for this project. The off and on switch is on the bottom.

Remove the bottom and discard the batteries and battery holder. Believe it or not, anodized aluminum is not only resistant to chemical attack, it also forms a high resistant coating which does not conduct electricity well. This made it somewhat difficult to connect the negative terminal of the board. The LEDs are in parallel and are connected to the spring. They have a voltage drop, so we do not want them in the circuit as it draws little power.

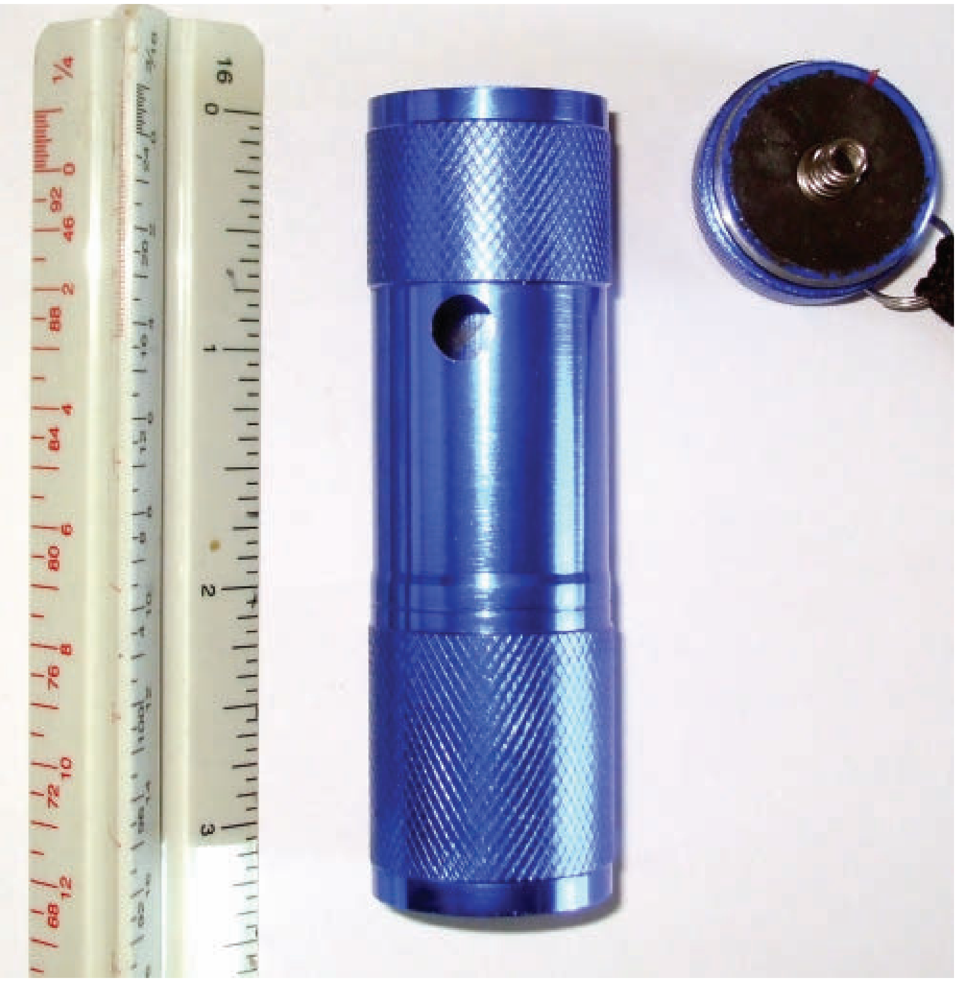
Using conductive glue, flood the LED board from the inside of the holder. Refer to **Figure 3**. This will short all the LED pins and connect it to the tube. It also acts as a ground shield against magnetic fields and noise. Remove the switch and drill a 1/4" hole 7/8" from the bottom for the speaker hole; refer to **Figure 4**.



■ FIGURE 3.

Building The Board

This whole project (including the board chassis and preprogrammed microprocessor) is available from the

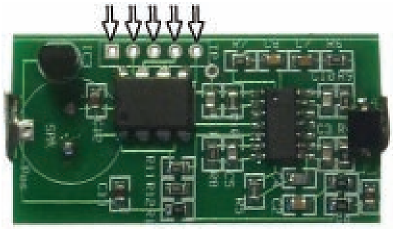


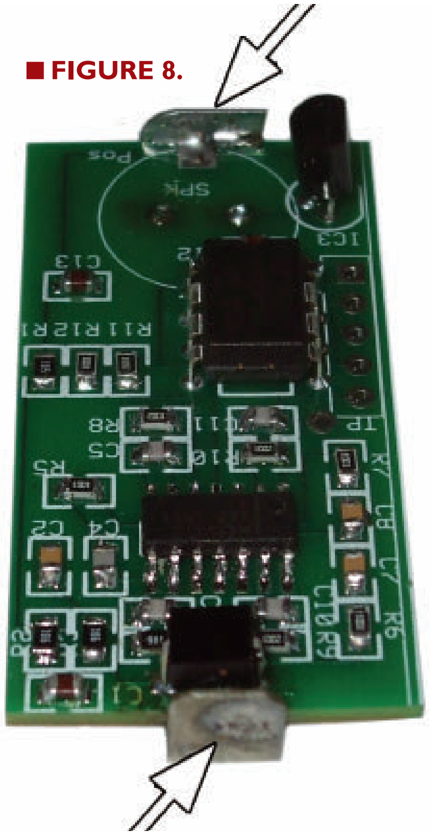
■ FIGURE 4.

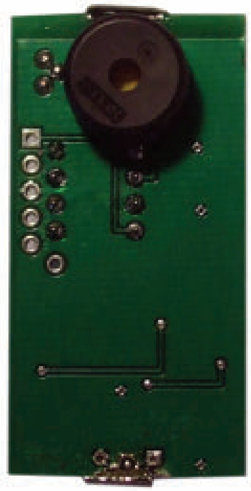
PARTS LIST

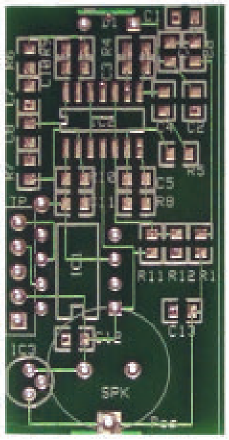
ITEM	DESCRIPTION	QTY	DIGI-KEY#
All surface-mounts are 805			
D1	BPW34FA	1 ea	475-1072-ND
C1-C12-C13	.1 µF	3 ea	445-7535-1-ND
C2-C7-C8	.01 µF	3 ea	445-1344-1-ND
C4	.001 µF	1 ea	445-1330-1-ND
C3-C5-C10-C11	100 pF	4 ea	445-1329-1-ND
IC1	PIC12F1822	1 ea	PIC12F1822-I/P-ND
IC2	MAX4478	1 ea	MAX4478ASD+-ND
IC3	LM7805	1 ea	497-2952-5-ND
R1	1 megohm	1 ea	541-1.0MACT-ND
R2-R3-R4	10 megohm	3 ea	541-10MACT-ND
R8-R9-R10-R11	10K ohm	4 ea	541-10KACT-ND
R5-R6-R7-R12	1K ohm	4 ea	541-1.0KACT-ND
Speaker	4 kHz, piezo	1 ea	445-2525-1-ND

Both the PICkit 3 and a kit of parts are available from the *NV* webstore at **https://store.nutsvolts.com**.


■ FIGURE 7.


■ FIGURE 8.


■ FIGURE 5.


■ FIGURE 6.

NV webstore if you're interested. The board layout was done with ExpressPCB free software. I have added a ground plane on the bottom of the board which helps eliminate noise; see **Figure 5**.

When dealing with ground planes, I think it's easier to solder from the top of the board which will help eliminate solder bridges on the IC3 and IC1.

Solder the surface-mount resistors and capacitors to their appropriate locations. I recommend using a magnifier, tweezers, and tacky flux. I normally use a long conical .01" tip and .025" rosin core solder.

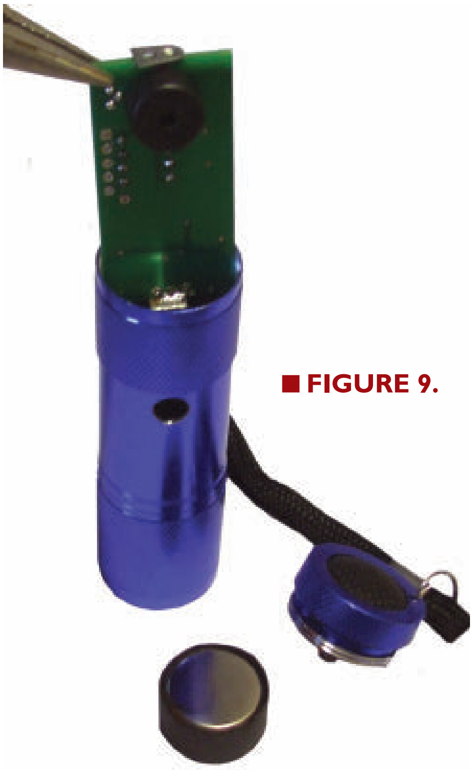
Solder IC2 noting its outline on the board. Tack one lead, and then solder the rest of the leads. If there are solder bridges, use solder wick to remove them.

Solder IC1. Lead 1 goes to

the square pad (**Figure 6**). The five holes on the left side are programming pads and nothing is soldered into them (**Figure 7**). Solder IC3 noting its flat.

The power connectors are made out of two flat spade terminals that are cut and soldered to the top and bottom of the board. Make sure you perform this before soldering the pin diode or the speaker as they will be in the way if soldered first; refer to **Figure 8**.

Locate the pin diode and look for the pin that has a cross. This is the cathode. Solder this pin to the square pad. Turn the board over and place the speaker into its hole. Solder from the top of the board (**Refer to Figure 5**). Check out the "Hints and Tips" sidebar for help with surface-mounts.


■ **FIGURE 9.**

Testing The Unit

Slide the board into the holder with the negative terminal toward the LEDs. Cut a piece of 3/4" shrink tubing to a length of 3/8". Note the + on each of the batteries. Place three CR 2032 coins into the tubing with the + up. Slide the battery assembly into the tube with the + toward the board and the negative side next to the spring (**Figure 9**). Screw in the bottom. Press the bottom switch to turn the unit on. Your “detector” will beep for about five seconds while its testing the battery. You should get about 10 clicks per minute.

Now go out there and radiate! **NV**

MAKE A SMART NECKLACE

By Craig A. Lindley
calhjh@gmail.com

Go to **www.nutsvolts.com/index.php?/magazine/article/july2013_Lindley** for any additional files and/or downloads associated with this article. You can also discuss this topic at **http://forum.nutsvolts.com**.

Our little town of Manitou Springs, CO takes its Carnival celebration seriously. It seems like half of the town is in the parade dressed in colorful and comic costumes, while the other half is on the streets partying and having a good time. Beaded necklaces of all varieties are traded between the participants and many people have quite the collections. I knew I had to come up with something special that no one else would have for our next Carnival, so I started thinking about making an electronic necklace of some kind. I wanted something that was unique, playful, personalized, and inexpensive so I could give it away if I wanted to, and wouldn't feel bad if it got broken during the celebration.

Having a basic idea of what I wanted, I searched the Internet and found **http://tinkerlog.com/howto/64pixels** where Alex Weber had connected a 64 LED array to a ATtiny2313 microcontroller to make a circuit capable of displaying static patterns, text strings, and simple animations. This was basically what I was looking for. Subsequent searches turned up another site (**https://sites.google.com/site/tinymatrix**), where tigeruppp (who also referenced Alex's work) built a 5x7 LED matrix and a ATtiny4313 into a necklace which was exactly what I wanted to do. Both Alex and tigeruppp provided C code for their projects, so I had a lot of resources to draw upon.

In the end, I used tigeruppp's hardware configuration and modified his code to build what I'll refer to as a smart necklace. So, in this article, I will show you how to:

1. Compile the C code for the smart necklace for the ATtiny4313 microcontroller
2. Program (Flash) the microcontroller using an Arduino Uno board as a programmer.
3. Build the smart necklace.

When we are finished, you will have a necklace with seven animations and 13 static display patterns available at the touch of a button. In addition, each necklace can be customized to display a person's name or other pertinent text message.

Prerequisites for building this smart necklace include:

1. Having a PC or MAC for compiling the code.
2. Having some knowledge of C program development; the ability to edit a C program source file and run a Makefile from a command line window, for example.
3. Having a programmer for Flashing the code into the controller (more on this shortly).
4. Having basic soldering skills for assembling the necklace.

After having built a few of these necklaces for my Mardi Gras friends, I realized they would also make great gifts for our young nieces and nephews; something they could show off to their friends. If you make these now, you will be way ahead when it comes to Christmas shopping, so let's get started.

The Hardware

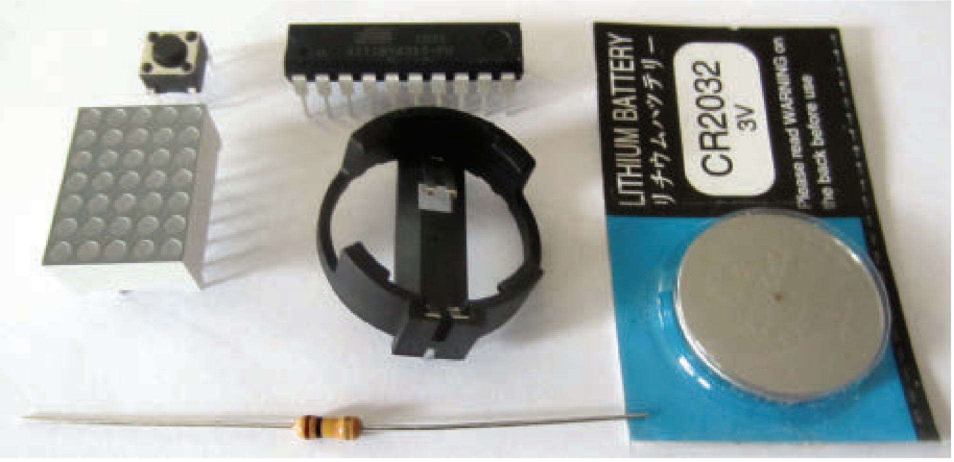
Figure 1 shows the Parts List for each necklace and where I bought everything. Each necklace can be made for around $5 and — if you buy the parts in quantity — even less. Once you have built a couple, you can assemble the necklaces in about half an hour. **Photo 1** shows the parts involved. **Figure 2** is the schematic of the necklace's electronics.

As you can see from the **schematic**, the circuit is very simple which makes it easy to build. What you cannot see from the **schematic** is that the middle 12 pins of the microcontroller map directly to the 12 pins on the LED matrix. When we build the necklace, we slide the LED matrix over the middle 12 pins of the controller and solder them in place. More on this later.

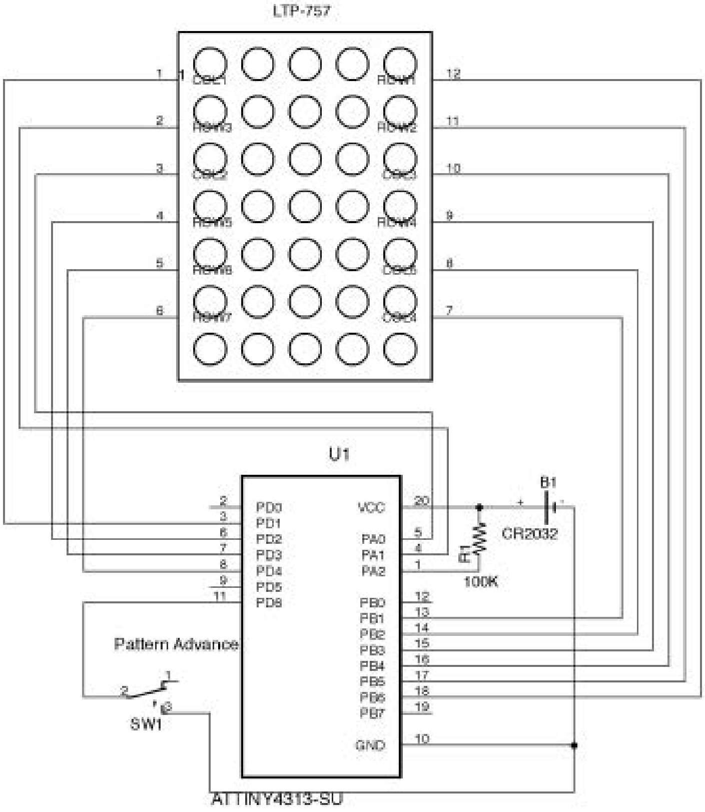
SMART NECKLACE PARTS LIST

ITEM	PART #	SOURCE / PRICE
U1	ATtiny4313 microcontroller	**digikey.com** - $1.51
LED1	LiteOn LTP-757 LED matrix	**jameco.com** - $0.89
R1	100K ohm 1/4 watt resistor	**evilmadscientist.com** / 10 for $0.40
SW1	Tactile button switch	**evilmadscientist.com** / 5 for $2.00
B1	2032 coin battery	Any local store / ~1.00
BH1	Coin battery holder	**jameco.com** - $0.79
N.A.	Hookup wire, heat shrink tubing	Any local store

■ **FIGURE 1.**



■ **PHOTO 1.** All of the required parts.



■ **FIGURE 2.** Smart Necklace Schematic.

The Programming Hardware And Software

When I decided to build these necklaces, I knew I would need a method of programming (a.k.a., Flashing) the ATtiny4313 chips. I knew an Arduino Uno (or other Arduino boards) could be used to program the chip by connecting the wires as shown in **Table 1**.

Arduino Uno Pin	ATtiny4313 Pin
13	19
12	18
11	17
10	1
+5V	20
GND	10

Table 1.

PROGRAMMING HARDWARE PARTS LIST

■ FIGURE 3.

PART #	PRICE
ISP Shield	$12.95
Arduino Uno (if you don't have one)	$29.95
ATtiny target board	$3.00
20-pin ZIF socket	$1.40
Six-pin DIL ribbon cable	$2.95
Six-pin DIL header	$0.70
USB male A to male B cable for connecting your computer to the Arduino Uno.	$2.95

All items listed here are from **evilmadscientist.com**

These connections can be made on a breadboard for the lowest cost approach. I, however, wanted a real Atmel device programmer that would be capable of Flashing Arduino bootloaders into new ATmega328 chips for other projects. I decided to buy the inexpensive ISP Shield kit and associated components from **evilmadscientist.com**. **Figure 3** shows the parts required and **Photo 2** shows the assembled programmer hardware.

The **evilmadscientist.com** site has the instructions for building the ISP Shield and the ATtiny target board. In this configuration, the ISP Shield is plugged into the Arduino Uno and the ISP Shield plugs into the ATtiny target board via a six-pin ribbon cable. ATtiny4313 chips are placed in the ZIF (Zero Insertion Force) socket on the target board and the lever is lowered to lock the chip in place during the programming process.

Whether you use the dedicated wiring or the ISP Shield approach, you will need to connect the Arduino Uno to your PC or MAC via a USB cable. You will also need to have the Arduino IDE installed on your computer. I used version 1.0.4 of the IDE. See **http://arduino.cc** for information about getting and installing the Arduino IDE if you haven't done so already.

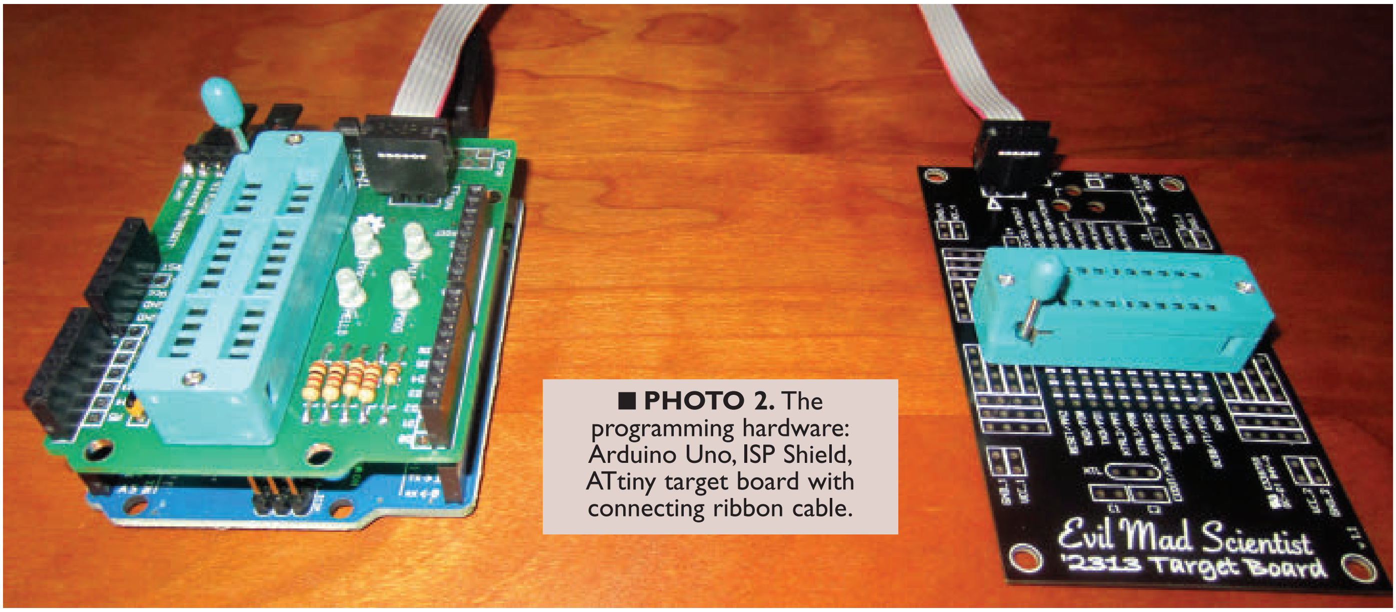
With the IDE installed and the USB cable connected, load the ArduinoISP sketch. Then, select the Arduino Uno from the IDE Tools/Board menu and click the upload button. If you are using the ISP Shield, the pulse LED will slowly Flash indicating you are ready to program the ATtiny4313. You can then close the Arduino IDE since the programming sketch has been loaded into the Arduino Uno. Actual programming of the ATtiny4313 will be done after we install the software development tools we'll discuss next.

Software Tools

You must install AVR software development tools on your computer to compile the smart necklace C code and to Flash the compiled code into the microcontroller chip. The controller chips are programmed before being assembled into a necklace.

CrossPack (**www.obdev.at/products/crosspack/index.html**) is the development environment for Atmel's AVR microcontrollers running on Apple's OS X and is similar to WinAVR (**http://sourceforge.net/projects/winavr/**) for Windows. These free development environments consist of a GNU compiler suite, a C library for the AVR, the avrdude uploader/programmer, and several other useful tools. Install the development tools for your computing environment as described on the appropriate website.

At this time, also download the file smartnecklace.zip from the article link and unzip it into a directory of your choosing. This zip file contains the files TinyMatrix.c (the C source code for the smart necklace) and a Makefile for



■ **PHOTO 2.** The programming hardware: Arduino Uno, ISP Shield, ATtiny target board with connecting ribbon cable.

building and programming the microcontroller.

The Makefile may need to be tailored to your computing environment. The Makefile I use on my Mac is shown here for reference. The line in bold is what will need to be changed if you are running on a PC. The *-P* command line switch specifies the serial port that will be used with the avrdude programmer.

On the Mac, the port is specified as */dev/tty.usb**; on a PC it would be changed to *COMx*, where *x* is the number of the serial port used by the programmer. The *-b* command line switch specifies the baud rate of the serial connection. Edit this file if necessary for your computing environment.

```
# Makefile for TinyMatrix
NAME       = TinyMatrix
OBJECTS     = $(NAME).o
DEVICE      = attiny4313
CLOCK       = 4000000
FUSES       = -U hfuse:w:0xDF:m -U lfuse:w:0xE2:m
PROGRAMMER = avrisp
AVRDUDE = avrdude -c $(PROGRAMMER) -p t4313 -P
/dev/tty.usb* -b 19200
COMPILE = avr-gcc -Wall -Os -DF_CPU=$(CLOCK) -
mmcu=$(DEVICE)
# symbolic targets:
all:   $(NAME).hex
.c.o:
       $(COMPILE) -c $< -o $@
flash: all
       $(AVRDUDE) -U flash:w:$(NAME).hex:i
fuse:
       $(AVRDUDE) $(FUSES)
clean:
       rm -f $(NAME).hex $(NAME).elf $(OBJECTS)
# file targets:
$(NAME).elf: $(OBJECTS)
       $(COMPILE) -o $(NAME).elf $(OBJECTS)
$(NAME).hex: $(NAME).elf
       rm -f $(NAME).hex
       avr-objcopy -j .text -j .data -O ihex
$(NAME).elf $(NAME).hex
```

With the Makefile properly configured, you will use the commands shown in **Table 2** from a command line window or terminal shell.

Command	Result
make	Compiles the C code in the file TinyMatrix.c. Use this command to find any errors in the source code. Errors will be displayed in the command/terminal window.
make fuse	Programs the ATtiny4313 fuses as required for operation of the code. This command only needs to be executed once per microcontroller chip.
make flash	Compiles the C code and if there are no errors, causes the compiled code to be Flashed into the microcontroller. The ATtiny4313 can be reprogrammed many times without issue. Perform this operation as many times as necessary to get the code correct.
make clean	Deletes all temporary object and hex files so you can continue with a clean slate.

Table 2.

Smart Necklace Code

The C code for the smart necklace is contained in the file TinyMatrix.c. This amazingly concise code was written by tigeruppp with a modification I did to allow the smart necklace to display text messages. You will have to study the code if you wish to understand its operation in detail, but here are some of the highlights:

- The controller runs on its internal clock at 4 MHz.
- The bitmap two-dimensional array in the code holds the data for the currently displayed frame.
- An interrupt service routine (ISR) drives the display process which is capable of updating the LED matrix approximately 78 times a second. (Fast enough to be flicker free.)
- The code polls the pattern switch between display updates. If a switch click is detected, the mode variable is updated which causes the next display pattern to be loaded into the bitmap array.

The code can be customized to display someone's name or a text message by changing the following line in the code:

```
// Set the following to the text you wish to
//display
#define TEXT “Hello”
```

So, if you wanted to change the displayed text from "Hello" to "Hello from Nuts and Volts Magazine" you would make this change:

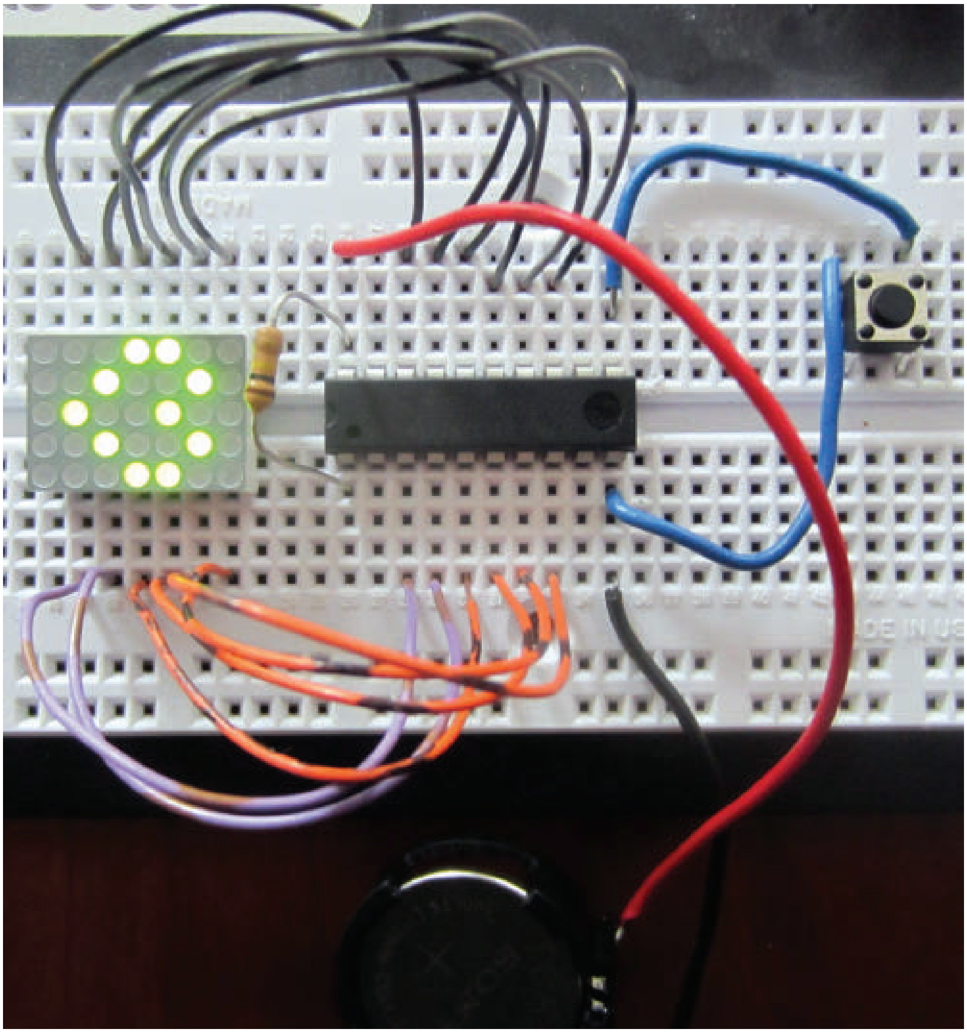
```
// Set the following to the text you wish to
//display
#define TEXT “Hello from Nuts and Volts
Magazine”
```

then rebuild the code with the "make flash" command. Now when the smart necklace is powered up, it will display this message over and over until the pattern switch is pressed to select a different display pattern.

If you are making a necklace for your nephew Bob, for example, you might change the code to:

```
// Set the following to the text you wish to
//display
#define TEXT “My name is Bob”
```

After you have made changes to the C code and have gotten it programmed into a ATtiny4313 microcontroller,



you might want to test it before building the necklace just to be sure your changes work as expected. I would suggest breadboarding the circuit as shown in **Photo 3** because once you build the ATtiny4313 into a necklace, it would be difficult to reprogram.

■ **PHOTO 3.** Breadboard of the smart necklace circuit.

Building A Smart Necklace

Now that we've got all the background information out of the way, it's time to actually build this thing. Please refer to **Photos 4**, **5**, and **6** during the following discussion.

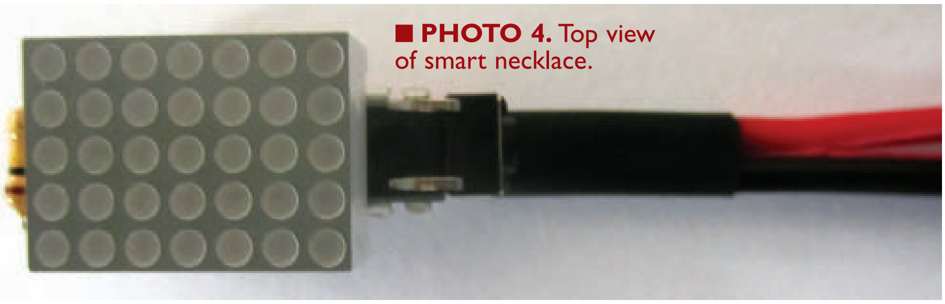
First, locate pin 1 of the ATtiny4313. It is marked with a small dot and a small indentation at the top of the DIP package. Next, locate pin 1 of the LED matrix which isn't obvious because there is no marking I could find. Pin 1 is on the top left of the matrix when the part number is to the right and you are looking at the top of the device.

Gently spread the leads of the LED matrix outward slightly. With both pin 1s on the left, slide the matrix onto the controller so that pin 1 of the matrix contacts pin 3 of the microcontroller. In other words, the LED matrix connects to the middle 12 pins of the microcontroller. Push the LED matrix down as far as it will easily go and solder one of the pins from the matrix to the microcontroller to hold it in place. When you are satisfied with the placement of the matrix, solder the remaining 11 pins to the controller chip.

Cut two pieces of light gauge wire for the necklace to around 16-1/2″ in length. I happened to have some red and black wire which made keeping the polarity straight easy. Position the 100K resistor on top of the controller chip and bend its leads so as to connect it between pins 1 and 20. Solder the resistor to pin 1. Connect the positive necklace wire — which, in my case, was red — to pin 20 as well, and solder it. Position the pattern display pushbutton switch at the opposite end of the chip and splay its leads so as to make contact to pins 10 and 11 of the microcontroller. Solder pin 11. Connect the negative wire (black, in my case) to pin 10 and solder.

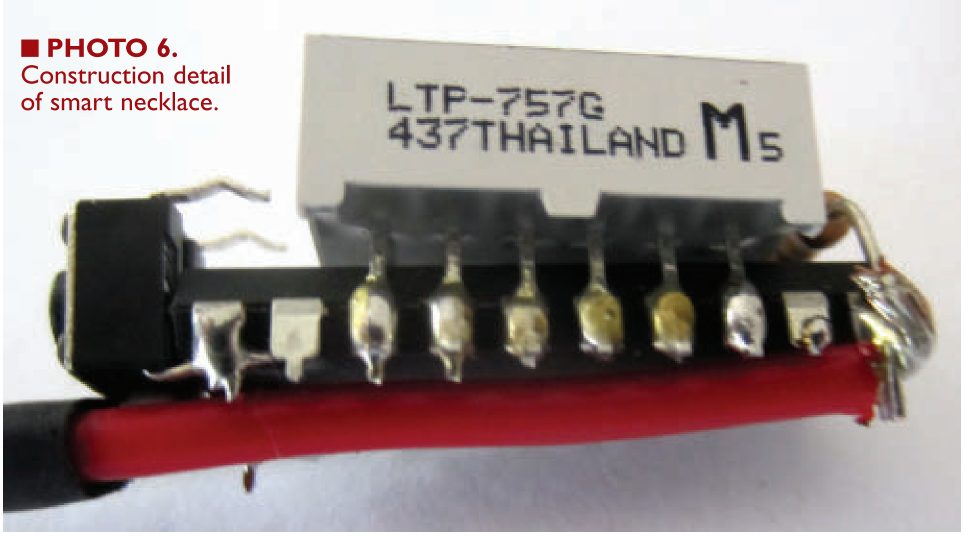
I then slid a 3/4″ piece of heat shrink tubing over the wires and pressed it up against the chip before shrinking it into place. You may want to trim the microcontroller pins at this time because they are awfully sharp.

I slid another 3/4″ piece of shrink tubing on the other end of the wires which I will shrink in a minute. As a strain



■ **PHOTO 4.** Top view of smart necklace.

■ **PHOTO 5.** Close-up of smart necklace.



■ **PHOTO 6.** Construction detail of smart necklace.

relief, I looped the wires through the battery holder before soldering them to the appropriate terminals. Note the polarity.

Next, I used some hot glue to anchor the wires to the battery holder. After this, I pushed the shrink tubing up against the battery holder and shrank it. With that, when you pop in the battery you should have an operational smart necklace. Click the pattern select switch to run through the various display patterns and animations that are available.

Craig has been interested in the production of lights/sounds/music with computers for a very long time. His most recent book is *Digital Audio with Java*, published by Prentice-Hall. He lives in the mountains of Colorado. When not messing around with electronics, computer projects, wood working, or beer brewing, he does a solo musical act around Colorado Springs.

Possible Enhancements

Since there is quite a bit of memory left in the controller, many more static display patterns and/or animations could be added once you understand how the code works. The code could also be altered to display different text messages with every press of the button.

Instead of the necklace configuration, the circuitry could be housed in small round or square boxes and used as Christmas tree decorations. Or, it could be sewn into the fabric of a shirt, vest, or coat. Or, it could also be used as an intelligent name tag displaying a person's name, company, and job title. If you think about it, many other uses will come to mind, as well. **NV**

By Rodney A. Colton
rodneycolton@verizon.net

Discuss this topic at **http://forum.nutsvolts.com**.

Using a Circa 1950 Rock-ola Jukebox Amp With CD Players and iPhones

This system plays my collection of vintage rock and roll and country music songs the way I remember how they used to sound.

When I was in high school in the early 1960s, I learned electronics with vacuum tubes. One of my favorite gadgets was the amplifier out of an old jukebox (I don't remember the make or model). I had a huge amount of fun and learned a lot — including what 400 volts DC felt like as it traversed my right hand, wrist, and forearm.

In mid 2012, I noticed an amplifier on eBay that was out of a 1952 Rock-ola 1436 jukebox, and just had to have it. It was essentially a dead carcass, as you can see in **Photo 1**.

Photo 2 shows the whole

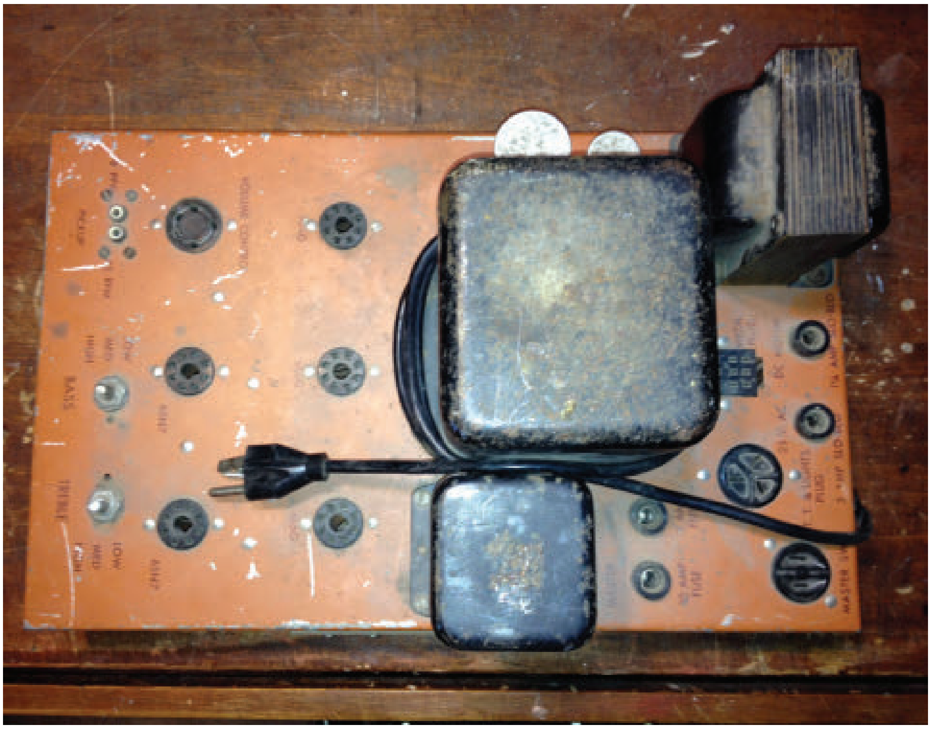


Photo 1. Rock-ola 1436 jukebox amplifier, as acquired.

Photo 2. A complete Rock-ola 1436 jukebox, circa 1952/53.

jukebox, similar to ones I remember existing in bars, cafés, and pool halls. **Photo 3** shows a 1436 amplifier inside a jukebox with its interconnecting control cables, speaker, etc.

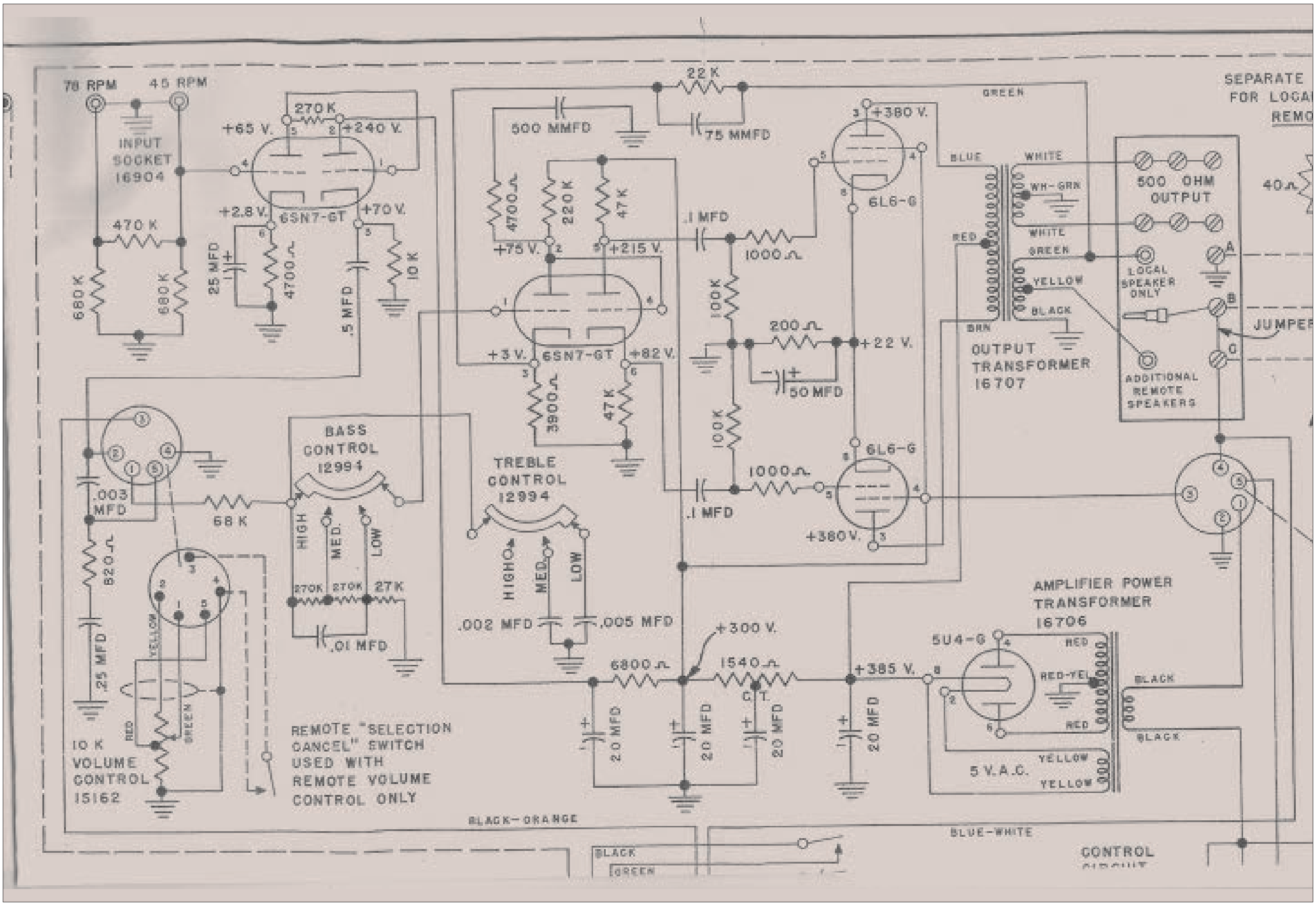
Purchasing an entire restored and working jukebox today is possible but was way outside my hobbyist budget, so I decided to restore the amplifier and make it able to stand alone. I wanted to make the restored amplifier as authentic as possible to the 1952 version.

Photo 3. The amplifier inside the jukebox.

I quickly determined that the transformers were still good, and decided to proceed. I acquired a complete service manual for a 1436 jukebox that came with a complete **schematic** (included here for reference purposes only) with all the tube pin voltages and parts list.

I decided that rather than just bringing the amplifier back to life with all of its scratches and rust, that I would strip it down to the bare metal and totally rebuild it using some of the existing electronic components, as well as any period-appropriate replacements. To help ensure that I could faithfully rebuild the internal electronics, I took several close-up photos like the one in **Photo 4**.

I knew from experience that after 60 years, all of the electrolytic capacitors and wax paper and foil capacitors would need to be replaced. I didn't have too much trouble finding exact replacements for the electrolytic capacitors, but the wax paper and foil capacitors had to be replaced with somewhat more modern disk ceramic or



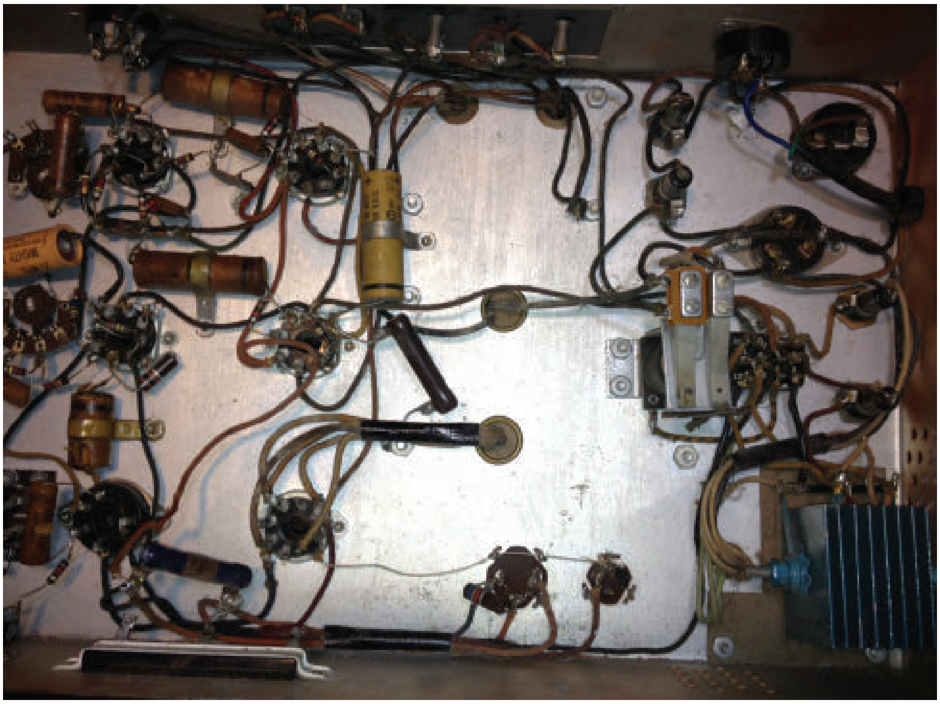


Photo 4. Reference shot of the underside of the 1436 chassis before restoration.

polypropylene film capacitors.

The relay and selenium rectifier that you can see in **Photo 4** were part of the control circuitry for other jukebox mechanisms. **Photo 5** shows the chassis after removal of all the components. I found it advantageous to remove most of the electronics as one mesh (seen on the right side of **Photo 6**).

This greatly assisted me during the rebuild in the positioning of components and the routing of wires in order to help ensure that 60 Hz hum and other problems with the layout could be minimized. I had decided to use new (but period-specific) tube sockets and new resistors in most instances. You can see the transformers getting their facelift with fresh paint.

Photo 7 shows the finished chrome-plated chassis. I decided not to attempt to fill or otherwise disguise any unused holes in the chassis, but to fill the holes with their original connectors — even if they were unused in the rebuilt amplifier.

There actually aren't any dents or blemishes in the chassis. What shows in **Photo 7** are reflections from other surfaces in the room where the photo was taken.

Over the span of a couple months at the rate of a few hours here and there, evenings and weekends, I tracked down all of the parts I needed. For harder-to-find items — such as tube sockets, tubes, electrolytic and high voltage capacitors — my best source was Antique Electronic Supply (**www.tubesandmore.com**). The vacuum tubes are all new Russian-made varieties with the original tube numbers. It's nearly impossible to describe the precise sequence of re-assembly and testing of all the electronics, but naturally great care was taken to position components and route wires as closely as possible to the original positions.

The first major milestone was mounting and testing all of the transformers. I was very glad that I had marked each wire on each transformer. Even though all of the wires were originally different colors, after 60 years they were all pretty much the same dull brown, not to mention a little brittle. This stage included installing the power cord, main fuse holder, and a main power switch (in one of the unused fuse holder holes).

The next major phase included routing most of the wires, such as the filament and high voltage wires. I found that while the transformer wires were a little brittle, the other original cloth-coated wire from 60 years ago was still fully usable. Nevertheless, I used new (but again, period-appropriate) cloth wrapped

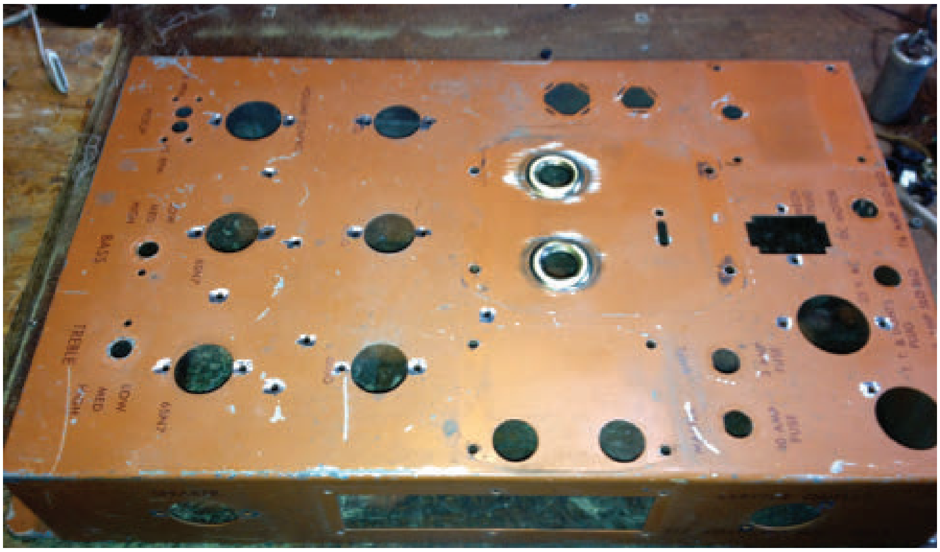


Photo 5. Bare chassis before refinishing.

plastic insulated wire throughout for authenticity.

I made another significant change to the wiring layout. My original design had one leg of the 6.3 VAC filament voltage grounded, with only one wire carrying the filament voltage to each tube. This method is a known source of 60 Hz hum problems, and was not used in subsequent models of this and other early vacuum tube amplifiers. Therefore, I used tightly twisted pairs of wires to carry the filament voltage to each tube.

I chose the routes for the twisted pairs using known good techniques described in old handbooks and the routing from the original design. Newer models of this amplifier and others from the period also included a "hum balance" control consisting of the potentiometer connected between the two legs of the 6.3 VAC voltage, with the sliding tap connected to ground. It turned out to be unnecessary.

As the wiring proceeded, when any live testing was conducted I always brought up the amplifier using a variac while checking numerous voltages throughout the circuit. This was mostly necessitated by two facts.

First, the 1950's distribution voltage was 110 VAC, while most distribution voltage today is usually 120 VAC or more. Mine consistently measures 123 VAC. The second reason is that the original jukebox used a speaker which had a large electromagnet for its field coil that drew a significant amount of high voltage current from the amplifier's high voltage power supply. **Photo 8** shows the completely rebuilt amplifier.

You can also see in the lower part of **Photo 8** several large power resistors in series/parallel combinations whose values were mostly experimentally determined to

Photo 6. Repainted transformers and underside wiring assembly.

Photo 7. The 1436 chrome-plated chassis ready for rebuild.

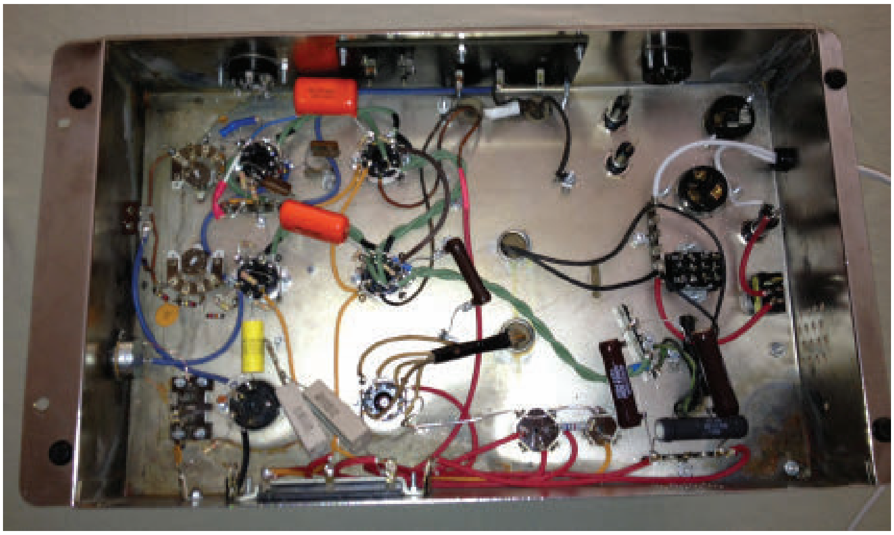
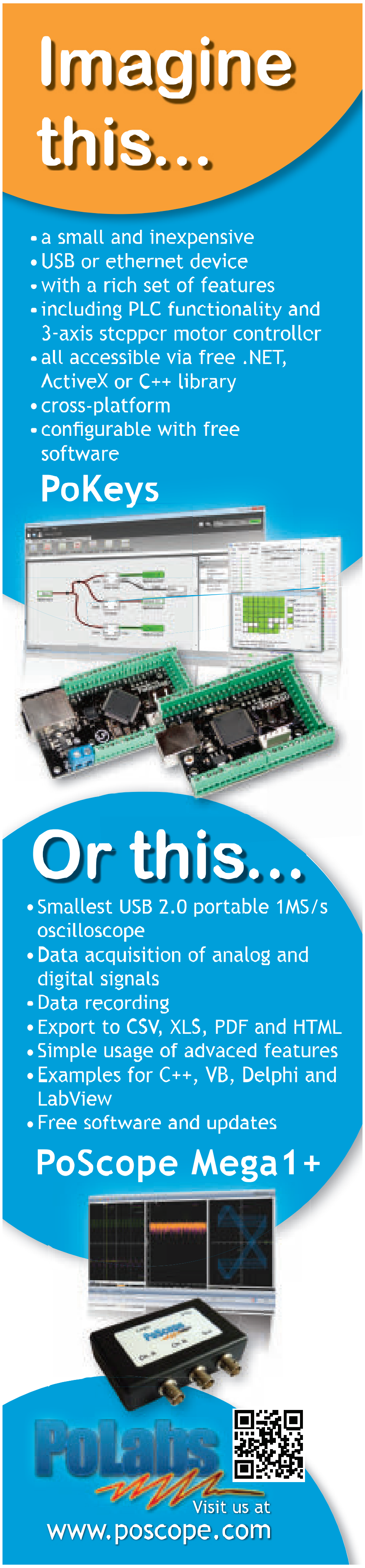


Photo 8. Completely rebuilt amplifier.



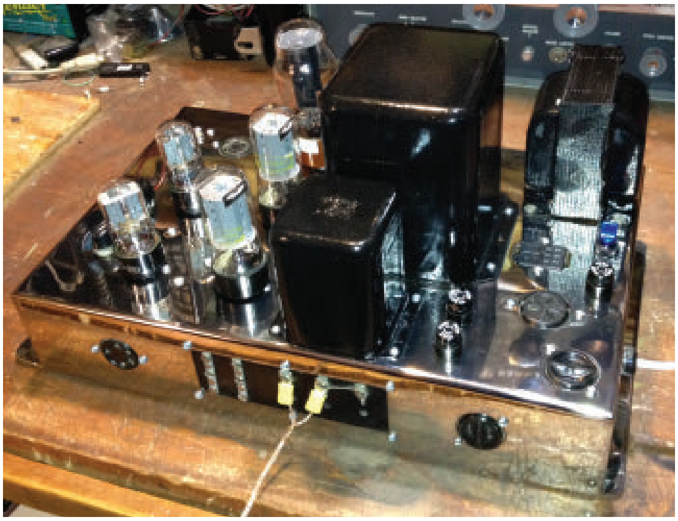
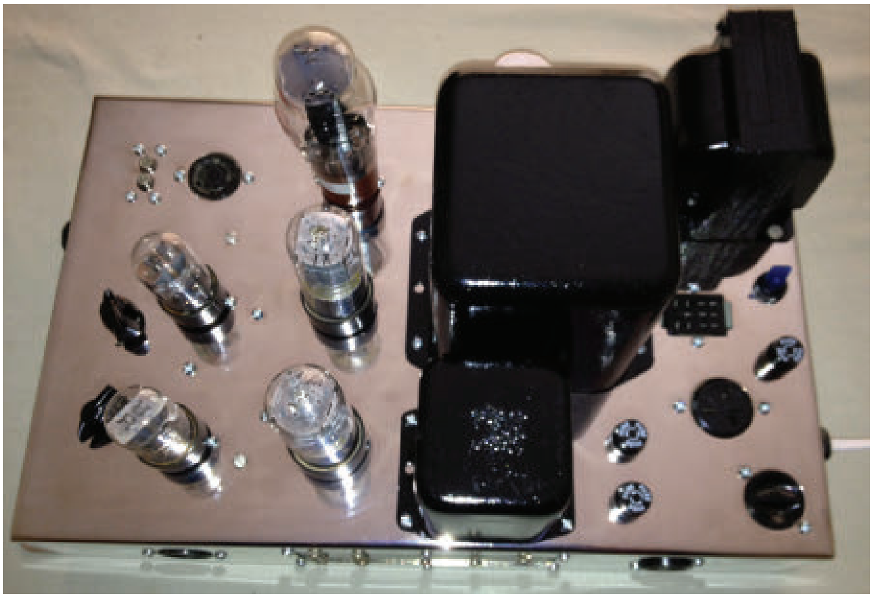


Photo 9. The completely restored jukebox amplifier.

Photo 10.



compensate for the lack of current drain by a large speaker field coil and the slightly higher line voltage. You may also notice that several sockets were simply cleaned and reinstalled to avoid the presence of unsightly empty holes in the chassis.

Since the original amplifier was installed deep inside of the jukebox, it had an external volume control. So, I installed a volume control in the chassis on the left. That was the only new hole that I drilled. **Photos 9** and **10** show the completed project.

I believe that the restored amplifier is as close to authentic 1950's technology as can be done in 2013. The chrome-plated chassis might be a bit of overkill, but it is a thing of beauty!

I was pleasantly rewarded with practically inaudible 60 Hz hum and perfect sound (well, as good as you could get from a 1950's jukebox). I have it connected to a vintage 15 inch, eight ohm speaker and use various audio sources such as a CD player and my iPhone. All together, the system plays my collection of vintage rock and roll and country music songs the way I remember them. Elvis sounds like he did 50 years ago.

It does a pretty good job with newer music, as well. In spite of the fact it only produces about 20 to 25 watts of audio, it can still drive you from the room. **NV**

Jaycar Electronics

Better, More Technical

www.jaycar.com
1800 784 0263
July 2013

DIY Kits for Electronics Enthusiasts

Ultrasonic Antifouling Kit for Boats

Marine growth electronic antifouling systems can cost thousands. This project uses the same ultrasonic waveforms and virtually identical ultrasonic transducer mounted in a sturdy polyurethane housing. By building it yourself you save a fortune! Standard unit consists of control electronic kit and case, pre-built ultrasonic transducer, gluing components and housing. The single transducer design of this kit is suitable for boats up to 10m (32ft); boats longer than about 14m (45ft) will need two transducers and drivers. Basically all parts supplied in the project kit including wiring.

- 12VDC
- Suitable for power or sail
- Could be powered by a solar panel/wind generator
- PCB size : 78 x 104mm

$179.50*

HOT SELLER!

Cat. KC-5498

Now available Pre-built:
Dual output, suitable for vessels upto 14m (45ft) **YS-5600 $562.00**
Quad output, suitable for vessels upto 20m (65ft) **YS-5602 $749.50**

YS-5600

YS-5602

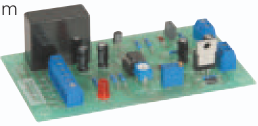
AUTOMOTIVE KITS

Universal Voltage Switch Kit

A universal module suits a range of different applications. It will trip a relay when a preset voltage is reached. Can be configured to trip with a rising or falling voltage making it suitable for a wide variety of voltage outputting devices eg. throttle position sensor, air flow sensor, EGO sensor. It also features adjustable hysteresis (the difference between trigger on/off voltage), making it extremely versatile. You could use it to trigger an extra fuel pump under high boost, anti-lag wastegate shutoff, and much more.

- Kit supplied with PCB, and electronic components.
- PCB size: 105 x 60mm

Cat. KC-5377



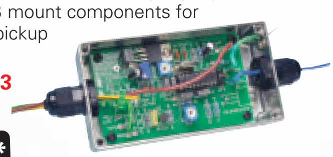
$23.75*

High Energy Ignition Kit for Cars

Use this kit to replace a failed ignition module or to upgrade a mechanical ignition system when restoring a vehicle. Use with virtually any ignition system that uses a single coil with points, hall effect/lumenition, reluctor or optical sensors (Crane and Piranha) and ECU. Features include adjustable dwell time, output or follow input option, tachometer output, adjustable debounce period, dwell compensation for battery voltage and coil switch-off with no trigger signal.

- Kit supplied with silk-screened PCB, diecast enclosure (111 x 60 x 30mm), pre-programmed PIC and PCB mount components for four trigger/pickup options

Cat. KC-5513



$36.00*

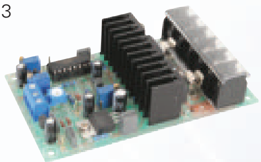
12/24VDC 20A Motor Speed Controller Kit

Control the speed of 12 or 24VDC motors from zero to full power, up to 20A. Features optional soft start, adjustable pulse frequency to reduce motor noise, and low battery protection. The speed is set using the onboard trimpot, or by using an external potentiometer (available separately, use **RP-3510 $1.50**).

- Kit supplied with PCB and all onboard electronic components
- Suitable enclosure UB3 case **HB-6013 $2.50** sold separately

Cat. KC-5502

$29.00*



12V Reversible Gearhead Motor

Works equally well in forward or reverse motion. For heavy duty applications.

- Massive 50kgs torque at 55RPM

Cat. YG-2738

$25.00*

See online for range of DC motors and stepper motors

Theremin Synthesiser Kit MkII

Create your own eerie science fiction sound effects by simply moving your hand near the antenna. Easy to set up and build. Complete kit contains PCB with overlay, pre-machined case and all specified components.

- PCB size: 85 x 145mm

Cat. KC-5475

$54.00*

For more details on each kit see our website

Arduino Experimenters Kit

Servo motor, lights, buttons, switches, sound, sensors, breadboard, wires and more are included with a Freetronics Eleven Arduino compatible board in this extensive hobby experimenter and starter kit.

HOT SELLER!

- Comprehensive instructions included
- Size: 340(W) x 165(H) x 36(D)mm

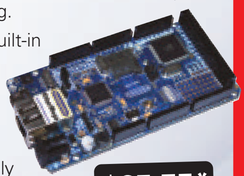
Cat. XC-4262

$65.00*

EtherMega, Mega sized Arduino Compatible Board with Ethernet

The ultimate network-connected Arduino-compatible board: combining an ATmega2560 MCU, onboard Ethernet, a USB-serial converter, a microSD card slot for storing gigabytes of web server content or data, Power-over-Ethernet support, and even an onboard switchmode voltage regulator so it can run on up to 28VDC without overheating.

- 10/100base-T Ethernet built-in
- 54 digital I/O lines
- 16 analog inputs
- MicroSD memory card slot
- Prototyping area
- Switchmode power supply



Cat. XC-4256

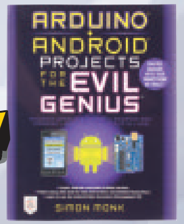
$85.75*

See online for extensive range of boards, shields and modules

Arduino + Android Projects for the Evil Genius

A project oriented book written around Android-type mobile phones and the Arduino microcontroller. Many projects use both the Arduino and the Android culminating in the creation of a home Automation system.

- Soft cover, 197 pages
- 215 x 275mm



NEW 2013

Cat. BM-7135

Also available: Arduino Robot Builder's Bonanza

- Soft cover, 464 pages

BM-7136 $31.75 NEW

$30.00*

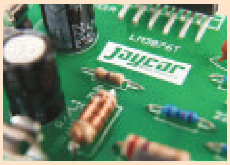
Don't Just Sit There...Build Something!

ATTENTION KIT BUILDERS

Can't find the kit you are looking for?
Try the Jaycar Kit Back Catalog

Our central warehouse keeps a quantity of older and slow-moving kits that can no longer be held in stores.

A list of kits can be found on our website. Just search for "kit back catalog".



HOW TO ORDER

PHONE:	1800 784 0263*
FAX:	+61 2 8832 3118*
EMAIL:	techstore@jaycar.com
POST:	P.O. Box 7172 Silverwater DC NSW 1811 Australia

*Australian Eastern Standard Time (Monday - Friday 6.30am - 5.30pm)
* US Eastern Standard Time (Monday - Friday 4.30pm - 3.30am)

Prices valid until 31/07/2013

*ALL PRICES IN USD & EXCLUDE POSTAGE & PACKING

NOW SHIPPING VIA DHL

5 - 10 day working delivery

• FAST DELIVERY • TRACK SHIPMENT

Note: Products are dispatched from Australia, so local customs duty & taxes may apply.

DHL

Reader Feedback

Continued from page 7

RPi. I also very much enjoy Ron Hackett's articles on the PICAXE microcontroller.

Richard Volk
Bismarck, ND

Thanks for the kudos Richard. Definitely stay tuned because we will have a second Pi article from Craig in an upcoming issue.

Bryan Bergeron

Parts Departed

I would like to know about Thomas Henry's use of a 1N4004 diode on his LCD in the March 2013 article detailing the DHT22 humidity sensor. On the schematic, there is a note saying to "see the parts list," but it was omitted from the article. Normally, I would choose a low value resistor to connect the LCD backlight to +5 volts. What is your reasoning behind using a diode?

Judy May W1ORO
Union, KY

Thanks for your interest in the DHT22 humidity sensor article.

Yes, the info about the rectifier was in the Parts List which didn't make it to the printed page, unfortunately. However, it is now available at the article link. The particular LCD I used (purchased through Microtivity/Amazon) specified the rectifier. On other units I've used in the past, a small valued resistor is specified. The purpose in either case is to drop a bit of voltage before the backlight LED, keeping the current flow to the LED at a safe level.

Use whichever the datasheet for your particular LCD requires.

By the way, the most recent surplus LCDs I got from All Electronics don't use anything at all. The device already has some sort of current limiting built in. For those, I just use a jumper.

So, this is one aspect that needs to be personalized to whatever LCD you are using. Hope that helps.

Thomas Henry

The Price Of Freedom

Bryan Bergeron's editorial comments were very well put in the June 2013 edition. When you are on a public street or downtown anywhere, you are not being private — not unless you have something to hide. Then, maybe it would be that person we are in need of finding.

I especially like the last paragraph and agree wholeheartedly. It reminds me that freedom is not free. Keep up the good work on the magazine. I like it.

Jim Inglett

Thanks for the note, Jim. Freedom is indeed not free.

Bryan Bergeron

To Censor Or Not To Sensor

I enjoyed "Developing Perspectives" in the June 2013 issue (The Duality of Technology), and felt compelled to throw my two cents in.

You make an interesting point about censorship. You cited a specific case about an overseas author writing about using a cell phone for a remote trigger. As we both know, a simple relay circuit could easily be altered to activate a detonator.

Should we then censor all articles with relays, or those with RF circutry, just because they COULD be used by bad people? I disagree with all censorship, but as an editor, I understand that you must adhere to certain standards.

I'm reminded of an argument for protecting our freedom of speech. I can't remember the exact quote, but it's something to the effect that "freedom of speech means having to endure hearing stuff I don't want to hear."

The problem in a nutshell is this:

1) There will ALWAYS be bad people trying to hurt others.

2) The electronics involved with an explosive device are pretty simple. You don't really need instructions from the Internet; a fuse can be lit by energizing an Estes igniter with a battery and a relay or a FET (or a BJT, for that matter!). In fact, you can do the same thing with a short length of nichrome wire wrapped around a fuse. It's not exactly rocket science. (Well, actually it is: model rocket science. That's something I learned about as a boy, about a gazillion years ago!)

3) Worldwide, ISPs and carriers don't give much thought to monitoring their subscribers.

I trust the existing laws governing explosives and flammable materials here, but I'm skeptical of laws in other countries.

Jeff Kerner

Thanks for the note, Jeff. It's a sticky issue. As you point out, it doesn't take much to create a trigger. However, someone outside of our field — with no electronics experience — might actually search for a cell phone trigger, simply because it's in the movies and in the news from the Middle East. But then, where do you draw the line? As a reader/viewer, I'm against most censorship. Just who is Big Brother? Some politician with an agenda? It will be interesting to follow the developments, politically and technologically.

Bryan Bergeron

More Pi, Please

I started a thread on the BYAP forum about the Raspberry Pi and I mentioned Ben Kibalo's article (see **www.backyardaquaponics.com/forum/viewtopic.php?f=50&t=16050&start=15#p387687**). People are really starting to talk about using the Pi for a monitor/controller.

Maybe folks will want more expandability out of Kibalo's article.

Bob Hassett

I'd like to hear more about your ideas. I also like your post on the aquaponics site. It sounds like you have a lot of thoughts about features you want for your specific system. Feel free to contact me — there are a lot of easy things to do with the RPi to add additional capability.

I think I might have another article for the August or September issue, but that will be after the growing season, so may be too late for your needs.

Let me know what your requirements are and perhaps I can help you out.

Ben Kibalo

BY FRED EADY

Accessing Bluetooth For Remote Control Action

I have a pretty good idea of how to interpret the bit patterns generated by the PlayStation DualShock 3's control array. However, I'm having too much fun writing the DualShock 3 driver firmware to care about button bit patterns right now. We have come a long way in a very short time. The first DualShock 3 discussion ended with us celebrating the ability to actually sense it via its USB portal. In the second column on this topic, we established a wired USB communications link with the DualShock 3 and enabled its internal logic. There is a Bluetooth radio embedded within the DualShock 3, and that's our next objective.

Go to **www.nutsvolts.com/index.php?/magazine/article/july2013_DesignCycle** for any additional files and/or downloads associated with this article. You can also discuss this topic at **http://forum.nutsvolts.com**.

PlayStation DualShock 3 Revelations

The PlayStation DualShock 3 communicates using the HID protocol. Information is passed between the DualShock 3 and PIC32MX using reports. In our case, a HID input report contains data that reflects the state of the DualShock 3's buttons, accelerometer, and joysticks. The HID input report generated also contains status information.

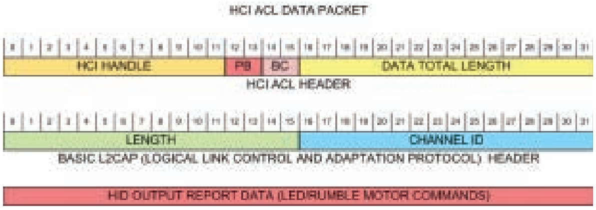
For instance, the charge state of the DualShock 3's internal battery is also part of the button status input report.

The buttons operated by the index fingers generate an eight-bit pressure value in addition to the one-bit pressed/released digital indication. Other pressure sensitive buttons include the up-down-left-right keypad on the left-hand side of the PlayStation unit. The up-down-left-right keypad's quad of buttons operates in a mutually exclusive manner.

The mechanics of the up-down-left-right keypad only allow one of the four buttons to generate a pressed/released action at any time. The symbol buttons (triangle, circle, square, and X) on the right-hand side of the DualShock 3 generate eight-bit pressure data that is identical to the index finger controls.

Unlike the up-down-left-right keypad, the symbol buttons are independent, with each button providing pressed/released data in addition to individual eight-bit pressure data.

The SELECT, START, and CONNECT buttons in the center portion of the controller do not generate pressure data. However, each of the aforementioned buttons provides one-bit pressed/released data. The pair of joysticks provides 16-bit position information. Each joystick also can be forced to generate one-bit pressed/released data. There are four user-accessible LEDs and a pair of rumble motors that are controlled by HCI ACL packets. **Figure 1** is a graphical representation of the HCI ACL packet used to influence the DualShock 3's LEDs and rumble motors. The payload of the HCI ACL packet looks like this:



■ FIGURE 1. Packets traveling via Bluetooth are composed of headers encapsulating other headers and data. This looks a lot like TCP/IP.

```
BYTE dualShockCmd[]
  = {
   0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00,
   0xFF, 0x27, 0x10, 0x00, 0x32,
   0xFF, 0x27, 0x10, 0x00, 0x32,
   0xFF, 0x27, 0x10, 0x00, 0x32,
   0xFF, 0x27, 0x10, 0x00, 0x32,
   0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00
  };
```

The *dualShockCmd* array is copied into the *HIDBuffer* array at initialization. A couple of HID header bytes are also added to the mix:

```
HIDBuffer[0] = 0x52; //HID
Bluetooth Set Report
   (0x50) | Report Type (Output
   0x02)
HIDBuffer[1] = 0x01; //Output
```

```
Report ID
for(i=0;i<48;++i)
       HIDBuffer[i+2] = dualShockCmd[i];
```

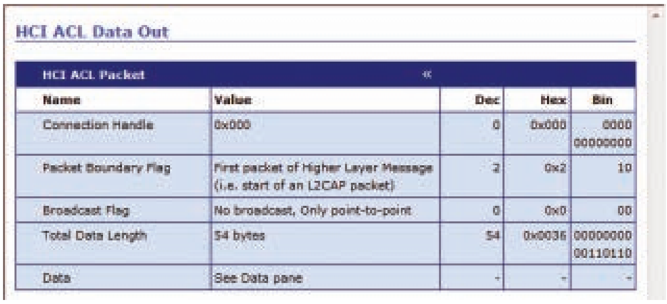
Once the payload array is in place, the HCI ACL packet is assembled and transmitted:

```
void setRumble(BYTE rightOnTime,BYTE
rightPower,BYTE leftOnTime,BYTE leftPower)
  {
  BYTE i;
  aclPktLen = 0x0036;
  l2capLen = 0x0032;

  hidcmdbuf[0]  = make8(hciHandle,0);
  hidcmdbuf[1]  = make8(hciHandle,1)  |  0x20;
  hidcmdbuf[2]  = make8(aclPktLen,0);
  hidcmdbuf[3]  = make8(aclPktLen,1);
  hidcmdbuf[4]  = make8(l2capLen,0);
  hidcmdbuf[5]  = make8(l2capLen,1);
  hidcmdbuf[6]  = controlCID[0];
  hidcmdbuf[7]  = controlCID[1];

  HIDBuffer[3]  = rightOnTime;
  HIDBuffer[4]  = rightPower;
  HIDBuffer[5]  = leftOnTime;
  HIDBuffer[6]  = leftPower;

  for (i = 0; i < 50; ++i)
       hidcmdbuf[8 + i] = HIDBuffer[i];

  do
  {
  USBTasks();
  }while(USBHostGenericTxIsBusy(deviceAddress));
  if((rc=USBHostGenericWrite(deviceAddress,
  hidcmdbuf,58)) != USB_SUCCESS)
       printf("\r\nRumble ON Command Failed");
  }
```

The *hciHandle* was obtained at connection time. The 0x20 value that is ORed with the *hciHandle* represents the Packet Boundary flag (PB/0x02) and the Broadcast flag (BC/0x00). The *aclPktLen* value is calculated beginning at *hidcmdbuf[4]*. The *l2capLen* value includes all bytes from *hidcmdbuf[8]* through the end of the packet.

The *make8* macro selects the high or low byte of a 16-bit word. The high byte selection is specified by entering a 1 as the second macro argument. Conversely, the low byte is selected by coding a 0 in the second macro argument.

Note that any previous contents of the *HIDBuffer* remain unchanged. We only alter the duration and power bytes associated with the left and right rumble motors. Once the coast is clear to transmit, we send the HCI ACL packet with a call to the Microchip Stack's *USBHostGenericWrite* function. Check out **Screenshot 1** which is a USB capture provided by the Ellisys USB Explorer 200. The data associated with **Screenshot 1** begins at offset 0x04 in **Screenshot 2**.

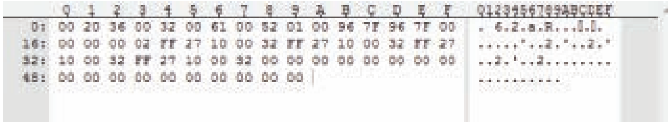


HCI ACL Data Out

HCI ACL Packet

Name	Value	Dec	Hex	Bin
Connection Handle	0x000	0	0x000	0000 00000000
Packet Boundary Flag	First packet of Higher Layer Message (i.e. start of an L2CAP packet)	2	0x2	10
Broadcast Flag	No broadcast, Only point-to-point	0	0x0	00
Total Data Length	54 bytes	54	0x0036	00000000 00110110
Data	See Data pane	-	-	-

■ SCREENSHOT 1. I've read sections of the Bluetooth specification over and over again. Nothing beats having the bits separated and explained using the Ellisys USB Explorer 200.

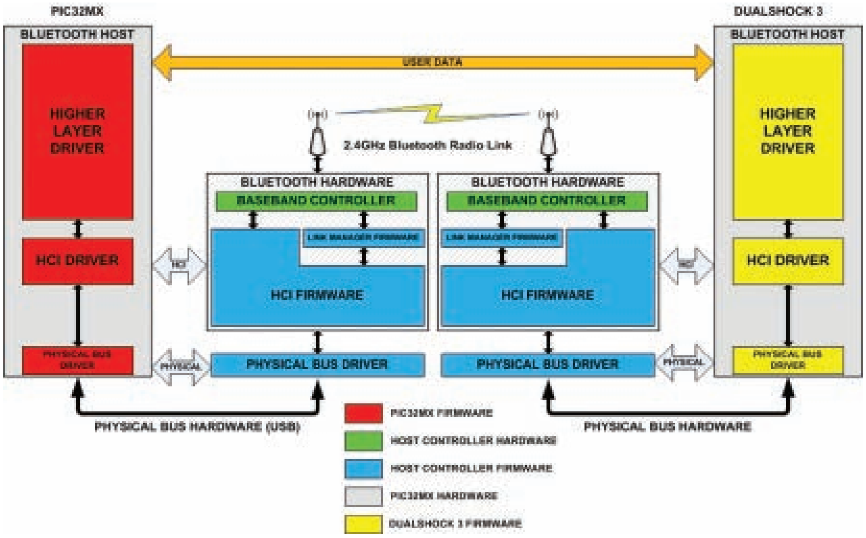


```
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
  0: 00 20 36 00 32 00 61 00 52 01 00 96 7F 96 7F 00   . 6.2.a.R.......
 16: 00 00 00 02 FF 27 10 00 32 FF 27 10 00 32 FF 27   .....'..2.'..2.'
 32: 10 00 32 FF 27 10 00 32 00 00 00 00 00 00 00 00   ..2.'..2........
 48: 00 00 00 00 00 00 00 00 00 00                     ..........
```

■ SCREENSHOT 2. Here is the data that goes with the Screenshot 1 HCI ACL packet capture.

HCI, ACL???

To understand how the DualShock 3 Bluetooth driver firmware works, you have to know how to speak the language of Bluetooth. I'll tell you right now that HCI is short for Host Controller Interface. ACL — or Asynchronous Connectionless Link — is the Bluetooth data link. ACL data travels over a packet-switched connection between two Bluetooth devices. The Bluetooth specification is very detailed and wordy. So, let's work our way through the Bluetooth stack as it relates to the PlayStation from bottom to top. We will begin by identifying the major assemblies you see in **Figure 2**.



■ FIGURE 2.The Bluetooth stack is a formidable creature. However, it is easily swallowed when eaten piece by piece.

The Bluetooth hardware depicted in the center of **Figure 2** is otherwise known as host controllers. The host controller is embedded in the Bluetooth hardware. On the PIC32MX side of the Bluetooth link, the host controller resides in the package under the lights in **Photo 1**.

The Bluetooth radio shown in **Photo 1** responds to the MotioninJoy Gamepad tool. You can make sure your Bluetooth radio will work with the PlayStation by running the Gamepad tool and attempting to pair the Bluetooth radio in question with your DualShock 3. If you can get the Gamepad tool Bluetooth pairing to work, you have a 99% chance that your Bluetooth radio will work with the PIC32MX Bluetooth driver firmware, as well.

The Cambridge Silicon Radio-based Bluetooth devices can be had from numerous eBay vendors. Be sure to select a radio that is described as Bluetooth V2.0 + EDR. Also, make sure that the Bluetooth radio's HCI and LMP (Link Manager Protocol) versions are at 3.0 or better. You can check the HCI and LMP versions using the Gamepad tool and a PC or laptop.

I figured you would want to see what was really inside that little dongle resting in **Photo 1**. So, I gave the Bluetooth radio some tequila and its clothes fell off in **Photo 2**. A sober (and fully clothed) Bluetooth radio is mounted on a chipKIT Network Shield in **Photo 3**.

The PIC32MX and DualShock 3 hardware support application firmware, data control firmware, and link control firmware. On the PIC32MX side of the Bluetooth piconet, the PIC32MX host communicates with its host controller via a USB connection. The DualShock 3's internal host electronics are also physically and logically connected to its embedded host controller.

The DualShock 3 internal interface may consist of any of today's digital data communications link such as UART, SPI, or I^2C. Since the internals of the PlayStation unit are transparent to us as far as writing the drivers are concerned, we don't really care what its host and host controller interface consists of.The Bluetooth radio is the lowest layer defined in the Bluetooth specification.

Bluetooth radios are frequency-hopping devices that operate in the 2.4 GHz ISM band. The baseband (physical) layer resides just above the Bluetooth radio layer. The baseband protocol is implemented as a link controller. The link controller — along with the Link Manager — performs link level tasks such as power control, link setup, and link configuration. The Link Manager uses Link Manager Protocol (LMP) to discover and communicate with other remote Link Managers.

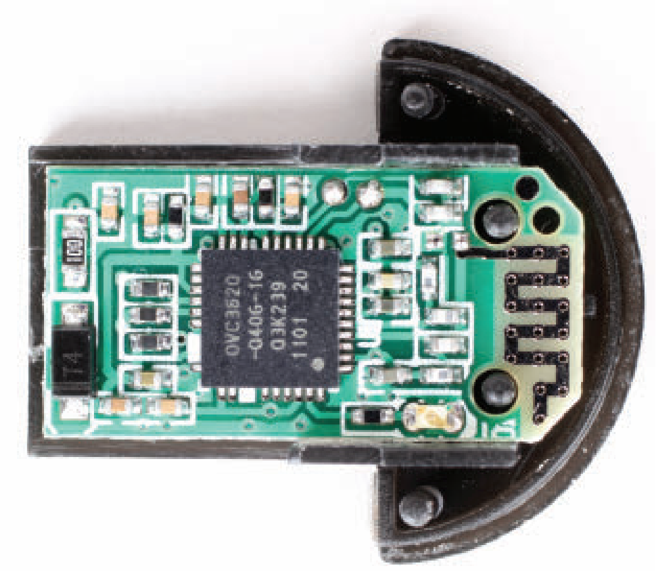
As you can see in **Figure 2**, the baseband controller portion of the host controller is physical hardware. In that the baseband controller is a hardware device, the baseband controller entity is most likely monitored and steered by a series of status and control registers. It is also evident that the baseband controller and Link Manager firmware have logical ties to the HCI.

The HCI is the command interface that gives us logical access to the baseband controller hardware registers and the Link Manager.

Note that the HCI spans both Bluetooth hosts and their associated controllers. The HCI firmware contained within the host controller implements HCI commands derived from baseband and Link Manager commands.

On the Bluetooth host side, the HCI driver firmware

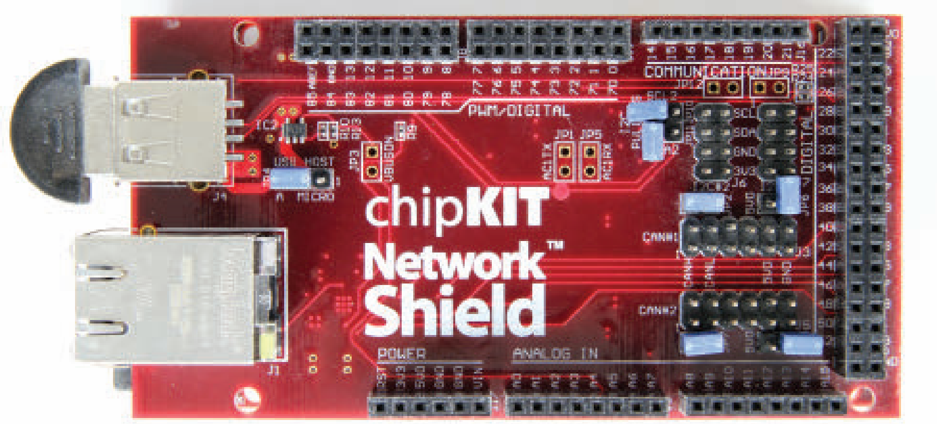
■ PHOTO 1. This particular Bluetooth device is equipped with a radio manufactured by Cambridge Silicon Radio. It's really cool that so many electronic components are housed in this tiny space.



■ PHOTO 2. The Bluetooth radio was easily disassembled using only a fingernail. The clock crystal and USB interconnects (etched and plated circuit board pads) are on the other side of the printed circuit board.

receives HCI event data from the host controller. The incoming HCI event data is parsed and acted upon accordingly by the Bluetooth host's HCI driver firmware. The communications link between the Bluetooth host's HCI driver and the host controller's HCI firmware is called the host controller transport layer. Official Bluetooth host controller transport layers include USB, UART, and RS-232.

The best way to explain L2CAP (Logical Link Control and Adaptation Protocol) is to walk around **Figure 3**. L2CAP is based on channels that transport control information and data between the upper layers of the Bluetooth stack using the baseband pipe. HID Feature reports always travel on the control channel. HID Input and Output reports use the interrupt channel. Note the inclusion of the control channel CID (Channel ID) in the HID LED command function:

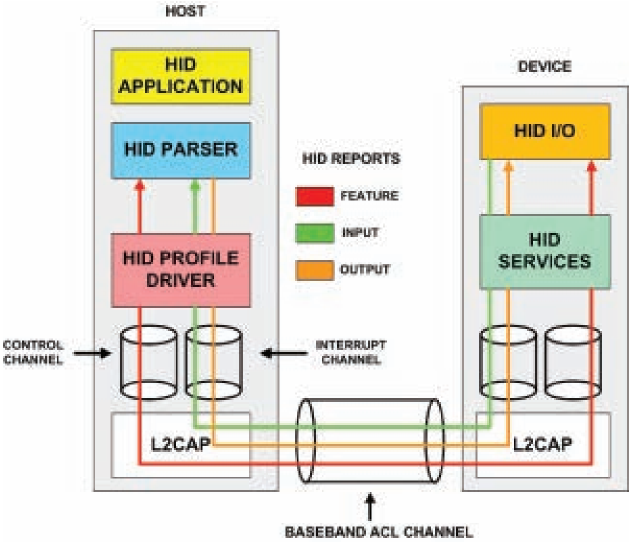


■ PHOTO 3. The compact size of the Bluetooth radio makes for a neat 32-bit Bluetooth remote control receiver package.

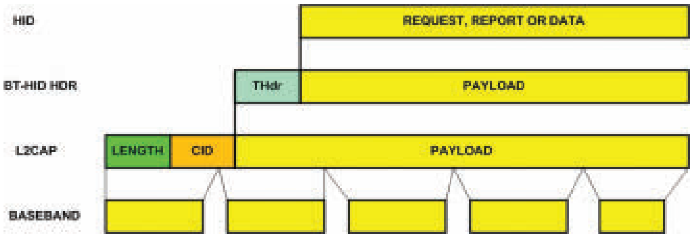
```
void ledOn(BYTE led)
{
     BYTE i;
     aclPktLen = 0x0036;
     l2capLen = 0x0032;

     hidcmdbuf[0] = make8(hciHandle,0);
     hidcmdbuf[1] = make8(hciHandle,1) | 0x20;
     hidcmdbuf[2] = make8(aclPktLen,0);
     hidcmdbuf[3] = make8(aclPktLen,1);
     hidcmdbuf[4] = make8(l2capLen,0);
     hidcmdbuf[5] = make8(l2capLen,1);
     hidcmdbuf[6] = controlCID[0];
               //Control Channel CID Low Byte
     hidcmdbuf[7] = controlCID[1];
               //Control Channel CID High Byte

     switch(led)
     {
     case 1:
        HIDBuffer[11] |= mLED1;
     break;
     case 2:
        HIDBuffer[11] |= mLED2;
     break;
     case 3:
        HIDBuffer[11] |= mLED3;
     break;
     case 4:
        HIDBuffer[11] |= mLED4;
     break;
     }

     for (i = 0; i < 50; ++i)
        hidcmdbuf[8 + i] = HIDBuffer[i];

     do
     {
        USBTasks();
     }while(USBHostGenericTxIsBusy(device
     Address));
     if((rc=USBHostGenericWrite(deviceAddress,
     hidcmdbuf,58)) != USB_SUCCESS)
        printf(“\r\nLED ON Command Failed”);
}
```

The control channel CID is also part of the HID *setRumble* function we discussed earlier.

Another way of looking at how L2CAP fits into the Bluetooth picture is shown in **Figure 4**. The HID portion of the packet could be incoming button data from the PlayStation. The incoming button data is tagged with a Bluetooth HID header. This header is used to route the payload data to the correct logical process in the upper layers of the Bluetooth stack. Since the HID payload needs to ride the L2CAP bus, an L2CAP header like the one described in **Figure 1** is fitted to the nose of the Bluetooth HID payload packet. L2CAP will automatically segment the packet as necessary and deliver it via the specified ACL channel to the remote destination.



■ FIGURE 3. L2CAP channels transport HID data and control information between the upper layers of the Bluetooth stack.



■ FIGURE 4. Use Figure 1 to separate the fields shown here into their assigned number of octets (bytes). This graphic is just one header short of an HCI ACL data packet.

Business As Usual

What I mean by “business as usual” is that the code behind the Bluetooth DualShock 3 driver is not very much different from code we would write to communicate using Ethernet, RS-232, or USB. The heart of this driver is parsing of the incoming ACL data generated by the DualShock 3. The results of the parsing operations drive state machines.

The HCI state machine rotates on these states:

```
typedef enum
{
  HCI_INIT = 0,         //perform Bluetooth Host
                        //device reset
  HCI_RESET,            //verify good reset
  HCI_GET_BLU_ADDR,     //get Bluetooth Host device
                        //address
  HCI_SCAN,             //make Host Bluetooth
                        //device visible
  HCI_INCOMING_CONNECT,       //handle incoming
                              //connect request
  HCI_GET_RMT_NAME,     //get ASCII name of remote
                        //device
  HCI_CONNECTED,        //handle connected status
  HCI_DISABLE_SCAN,     //make Host Bluetooth
                        //device invisible
  HCI_FINISHED,         //HCI processing complete
  HCI_DISCONNECT        //handle disconnect state
}HCI_State;
```

The HCI task processor monitors asynchronous HCI events. The contents of the incoming events determine the setting or clearing of flags. The state of the flags is taken into consideration by the HCI state machine, which performs HCI-related tasks according to the logic that stands behind the HCI flags. We will only need a subset of the full set of official HCI events to exercise the DualShock 3:

```
//************************************************
//*    HCI EVENTS
//************************************************
#define HCI_EVENT_CONNECTION_COMPLETE       0x03
#define HCI_EVENT_CONNECTION_REQUEST        0x04
#define HCI_EVENT_DISCONNECTION_COMPLETE    0x05
#define HCI_EVENT_REMOTE_NAME_REQUEST_
        COMPLETE                            0x07
#define HCI_EVENT_CHANGE_CONNECTION_LINK_
        KEY_COMPLETE                        0x09
#define HCI_EVENT_ROLE_CHANGE               0x12
#define HCI_EVENT_NUMBER_OF_COMPLETED_PKTS  0x13
#define HCI_EVENT_DATA_BUFFER_OVERFLOW      0x1A
#define HCI_EVENT_COMMAND_COMPLETE          0x0E
#define HCI_EVENT_COMMAND_STATUS            0x0F
#define HCI_EVENT_LOOPBACK_COMMAND          0x19
```

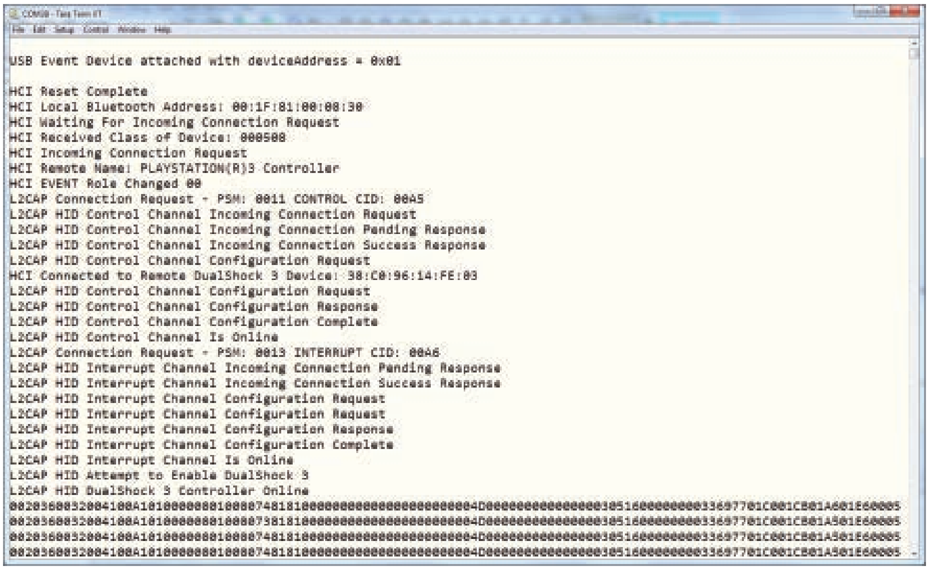
The L2CAP state machine functions on incoming L2CAP commands. The L2CAP commands are processed by the *l2capTasks* function:

```
//************************************************
//*    L2CAP (LOGICAL LINK CONTROL AND
//*    ADAPTATION PROTOCOL) STATES
//************************************************
typedef enum
{
   L2CAP_HOLD = 0,
   L2CAP_HID_CONTROL_CHANNEL_SETUP,
   L2CAP_HID_CONTROL_CHANNEL_REQUEST,
   L2CAP_HID_CONTROL_CHANNEL_ONLINE,
   L2CAP_HID_INTERRUPT_CHANNEL_SETUP,
   L2CAP_HID_INTERRUPT_CHANNEL_REQUEST,
   L2CAP_HID_INTERRUPT_CHANNEL_ONLINE,
   L2CAP_HID_ENABLE_DUALSHOCK,
   L2CAP_HID_INTERRUPT_CHANNEL_DISCONNECT,
   L2CAP_HID_CONTROL_CHANNEL_DISCONNECT,
   L2CAP_FINISHED
}L2CAP_State;
```

The PIC32MX Bluetooth driver source code is lengthy. So, we won’t attempt to cover every minute element of its firmware logic here. Instead, I have taken the liberty of adding code within the Bluetooth driver to alert us as to the status of the driver, using good old RS-232 and a terminal emulator.

Screenshot 3 is an RS-232 log of the startup USB, HCI, and L2CAP events. The initial USB event is the result of the Microchip USB stack querying the PIC32MX Bluetooth radio’s USB interface. The Microchip stack retrieves the Bluetooth radio’s descriptors and assigns a device address. Something has to kick off this USB stack, so here’s the code that wakes it up and manages the operation of the HCI and L2CAP task processors:

```
//************************************************
// MAIN
//************************************************
int main (void)
{
   BYTE i;
   init();           //initialize the chipKit
                     //hardware
   USBInitialize(0); //initialize the usb stack

   hciTimeOut = 0;   //initialize HCI firmware
                     //timer
   l2capState = L2CAP_HOLD; //initialize the
                            //L2CAP State
                            //Machine
   hciState = HCI_INIT;     //initialize the HCI
                            //State Machine
DoTasks = FALSE;            //preset task timer
flag
InitializeTimer();          //start the task
                            //timer
```



■ SCREENSHOT 3. This is a verbose description of the HCI and L2CAP operations that occur before and during a Bluetooth connection with the PlayStation DualShock 3.

```
USB Event Device attached with deviceAddress = 0x01

HCI Reset Complete
HCI Local Bluetooth Address: 00:1F:81:00:08:30
HCI Waiting For Incoming Connection Request
HCI Received Class of Device: 000508
HCI Incoming Connection Request
HCI Remote Name: PLAYSTATION(R)3 Controller
HCI EVENT Role Changed 00
L2CAP Connection Request - PSM: 0011 CONTROL CID: 00A5
L2CAP HID Control Channel Incoming Connection Request
L2CAP HID Control Channel Incoming Connection Pending Response
L2CAP HID Control Channel Incoming Connection Success Response
L2CAP HID Control Channel Configuration Request
HCI Connected to Remote DualShock 3 Device: 38:C0:96:14:FE:03
L2CAP HID Control Channel Configuration Request
L2CAP HID Control Channel Configuration Response
L2CAP HID Control Channel Configuration Complete
L2CAP HID Control Channel Is Online
L2CAP Connection Request - PSM: 0013 INTERRUPT CID: 00A6
L2CAP HID Interrupt Channel Incoming Connection Pending Response
L2CAP HID Interrupt Channel Incoming Connection Success Response
L2CAP HID Interrupt Channel Configuration Request
L2CAP HID Interrupt Channel Configuration Request
L2CAP HID Interrupt Channel Configuration Response
L2CAP HID Interrupt Channel Configuration Complete
L2CAP HID Interrupt Channel Is Online
L2CAP HID Attempt to Enable DualShock 3
L2CAP HID DualShock 3 Controller Online
00203600032004100A101000000001000007481810000000000000000000000004D000000000000000000305160000000003369770 1C001CB01A601E60005
00203600032004100A101000000001000007381810000000000000000000000004D000000000000000000305160000000003369770 1C001CB01A501E60005
00203600032004100A101000000001000007481810000000000000000000000004D000000000000000000305160000000003369770 1C001CB01A501E60005
00203600032004100A101000000001000007481810000000000000000000000004D000000000000000000305160000000003369770 1C001CB01A501E60005
```

```
    do{
        USBTasks();    //perform
usb stack

//housekeeping

        if(DoTasks == TRUE)
//trigger on task

//timer
        {
            DoTasks = FALSE;
//reset task timer

//flag
            hciTasks();
//run HCI task
                                //processor
            l2capTasks();       //run L2CAP task
                                //processor
        }
     }while(1);
}
```

The task timer fires every 10 mS. Thus, ideally, every 10 mS the HCI and L2CAP task engines run and influence the HCI and L2CAP state machines. The USB stack tasks are serviced with every pass of the *do/while* loop. According to **Screenshot 3**, the first official HCI task performed is the resetting of the Bluetooth radio plugged into the PIC32MX's Type A USB portal (see **Photo 3**). To give you some idea of how and what the PIC32MX HCI driver can do, upon completion of the HCI reset, I used an HCI command to retrieve the address of the PIC32MX's Bluetooth radio. The HCI state machine sits and waits until an L2CAP connection request is received from the DualShock 3.

Depressing the PlayStation's button will kick off a Bluetooth connection request. The Class of Device information was gleaned from addressing data generated by the resultant HCI Connection Request Event. Once an HCI connection was established between the PlayStation host controller and the PIC32MX host controller, I was able to call yet another HCI command to request and receive the DualShock 3's ASCII name.

To continue on with the L2CAP control and interrupt channel builds, the PIC32MX host must be configured as the master of the piconet and the DualShock 3 must be seen as the slave device. A successful HCI role change command is indicated in the **Screenshot 3** log (00 = Master Mode).

The DualShock 3 is a HID device, and we must prepare a communications path that is suitable for Bluetooth HID traffic. In other words, we need to set up a group of ACL channels on the host and the device. Take another look at **Figure 3**. The host control and interrupt CIDs are hard-coded in the PIC32MX Bluetooth driver:

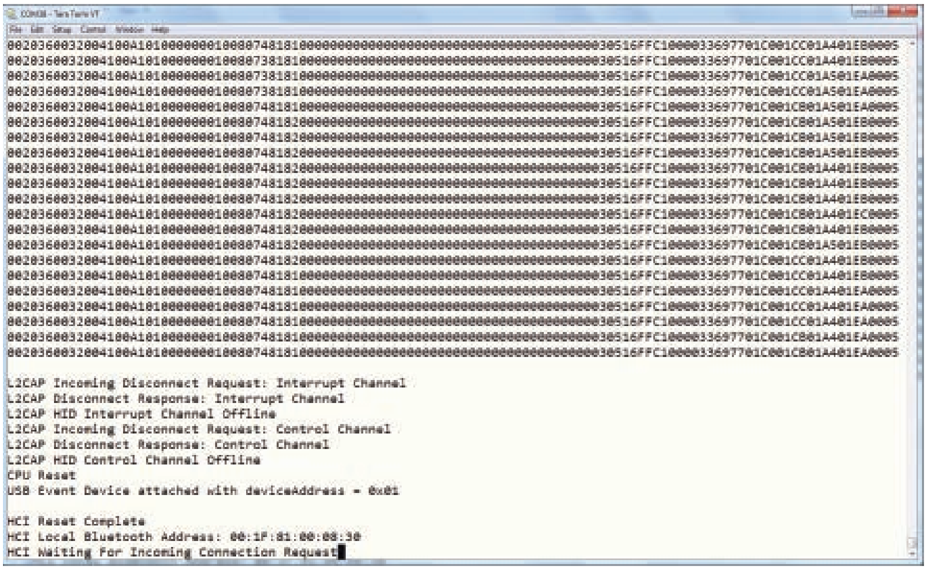
```
hostControlCID[0]  = 0x40;
hostControlCID[1]  = 0x00;
packedhostControlCID = make16(hostControlCID[1],
    hostControlCID[0]);
hostInterruptCID[0]  = 0x41;
hostInterruptCID[1]  = 0x00;
packedHostIntCID = make16(hostInterruptCID[1],
    hostInterruptCID[0]);
```

User assignable CIDs begin at 0x0040 and extend to 0xFFFF. I have yet to see the DualShock 3 use 0x0040 and 0x0041. So, it's pretty safe for us to use the very first available CID. The PlayStation will assign CIDs randomly during a connection request. I found that if you run one connect session after another, the CIDs are incremented by one for each successive cycle. The DualShock 3 CID memory allocation in the host driver looks like this:

```
BYTE controlCID[2];
        //L2CAP PlayStation DualShock 3 CID for
        //Control Channel
BYTE interruptCID[2];
        //L2CAP PlayStation DualShock 3 CID for
        //Interrupt Channel
WORD packedControlCID;
WORD packedInterruptCID;
```

The L2CAP entity is used to create the channels that will carry our Bluetooth HID packets. As you can see in **Screenshot 3**, the control channel is created and configured first. The PlayStation has informed us that its control CID is 0x00A5 and its interrupt CID is 0x00A6. Upon a successful connection, an HCI connected event is raised. I utilized this event to use HCI services to request the Bluetooth address of the DualShock 3.

With control and interrupt channels up and running, we can send the magic packet over the ACL pipe that gets the DualShock 3's attention. The **Screenshot 3** verbage tells us that the PlayStation was successfully enabled. I



```
0020360032004100A10100000000100080748181000000000000000000000000000000000000030516FFC1000033697701C001CC01A401EB0005
0020360032004100A10100000000100080738181000000000000000000000000000000000000030516FFC1000033697701C001CC01A401EB0005
0020360032004100A10100000000100080738181000000000000000000000000000000000000030516FFC1000033697701C001CC01A501EA0005
0020360032004100A10100000000100080738181000000000000000000000000000000000000030516FFC1000033697701C001CC01A501EA0005
0020360032004100A10100000000100080748181000000000000000000000000000000000000030516FFC1000033697701C001CB01A501EA0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CB01A501EB0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CB01A501EB0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CB01A501EB0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CB01A401EB0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CB01A401EB0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CB01A401EB0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CB01A401EC0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CB01A401EB0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CB01A501EB0005
0020360032004100A10100000000100080748182000000000000000000000000000000000000030516FFC1000033697701C001CC01A401EB0005
0020360032004100A10100000000100080748181000000000000000000000000000000000000030516FFC1000033697701C001CC01A401EB0005
0020360032004100A10100000000100080748181000000000000000000000000000000000000030516FFC1000033697701C001CC01A401EA0005
0020360032004100A10100000000100080748181000000000000000000000000000000000000030516FFC1000033697701C001CC01A401EA0005
0020360032004100A10100000000100080748181000000000000000000000000000000000000030516FFC1000033697701C001CC01A401EA0005
0020360032004100A10100000000100080748181000000000000000000000000000000000000030516FFC1000033697701C001CB01A401EA0005
0020360032004100A10100000000100080748181000000000000000000000000000000000000030516FFC1000033697701C001CB01A401EA0005

L2CAP Incoming Disconnect Request: Interrupt Channel
L2CAP Disconnect Response: Interrupt Channel
L2CAP HID Interrupt Channel Offline
L2CAP Incoming Disconnect Request: Control Channel
L2CAP Disconnect Response: Control Channel
L2CAP HID Control Channel Offline
CPU Reset
USB Event Device attached with deviceAddress = 0x01

HCI Reset Complete
HCI Local Bluetooth Address: 00:1F:81:00:08:30
HCI Waiting For Incoming Connection Request
```

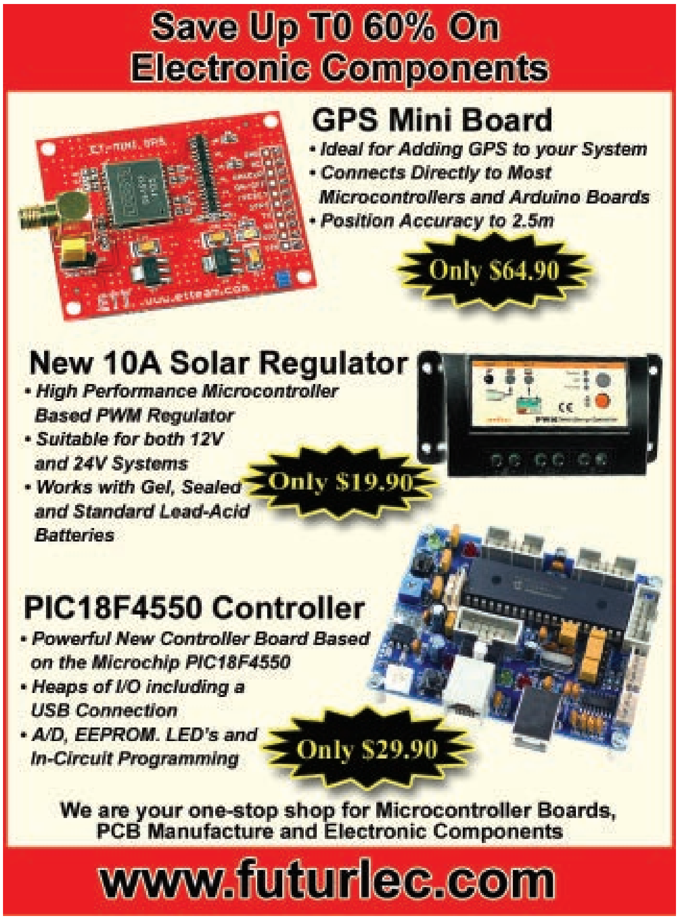
■ SCREENSHOT 4. Depressing PlayStation buttons for 10 seconds forces the DualShock 3 to issue disconnection requests.

chose to display the incoming 58-byte ACL data packet as readable ASCII characters represented in hexadecimal format. All of the DualShock 3's button state, button pressure, accelerometer, battery, and joystick data is contained with each 58-byte ACL data packet.

Stomping On The Brakes

The PlayStation button initiates the DualShock 3-to-PIC32MX Bluetooth communications session. The same button also disconnects the DualShock 3 and PIC32MX. Take a look at **Screenshot 4**. Depressing the PlayStation button for 10 seconds will force the DualShock 3 to issue an L2CAP interrupt channel disconnect request.

Upon receiving this disconnect request, the PIC32MX will issue a response. Once the interrupt channel is disabled, the control channel is disabled in the same manner. The PIC32MX Bluetooth driver issues a CPU reset command following the successful channel teardown process. The PIC32MX Bluetooth driver issues an HCI reset command and sits until it receives a new connection request.



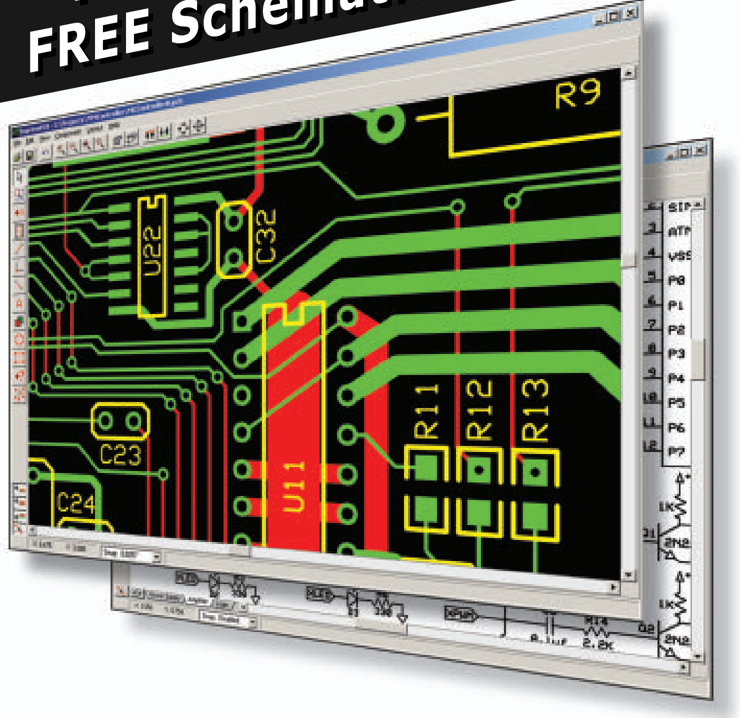
Next Stop

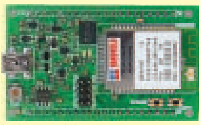
The PIC32MX Bluetooth driver code we've developed can also be ported to drive the Digilent MX4 cK and MX7 cK embedded mechatronic controllers. I'll be covering this soon in *Nuts & Volts* sister publication, *SERVO Magazine*. See you over there. **NV**

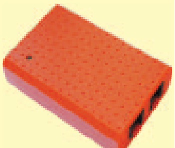
chipKIT MAX32
chipKIT Network Shield
MX7 cK, MX4 cK
Digilent, Inc. • **www.digilentinc.com**

MPLAB X, XC32 C Compiler
Microchip • **www.microchip.com**

Ellisys USB Explorer 200
Ellisys • **www.ellisys.com**







■ BY L. PAUL VERHAGE

LED Photometers Revisited — Part 2

Go to **www.nutsvolts.com/index.php?/magazine/article/july2013_NearSpace** for any additional files and/or downloads associated with this article. You can also discuss this topic at **http://forum.nutsvolts.com**.

Last time, we built an LED-based photometer that was better than the one discussed in this column back in 2007. Now, we need to test and calibrate it. One reason this is necessary is that the output of the LED is sensitive to its temperature. This is something that wasn't taken into account six years ago.

Gathering Data From The LED Photometer

Since we are planning to collect photometer data in near space, we need a flight computer to operate the photometer. If the photometer is to remain on the ground, then we can use a digital multimeter to measure its output.

Alternatively, we could automate the process of taking photometer readings on the ground by using the microcontroller circuit needed for a near space mission.

The flight computer connected to the photometer needs a program to digitize the two voltages produced by the photometer and store the results. Since my flight computers are based on PICAXEs, I will demonstrate using code samples specific to these microcontrollers. You will therefore need to adjust the code if you use something other than a PICAXE.

The process of converting analog voltages to their digital representation is called analog-to-digital conversion (ADC). The PICAXE has two Basic commands for doing this. Microcontrollers that can't do this themselves use special ICs to do it for them.

The first PICAXE ADC command converts a zero to five volt sensor output into an eight-bit value between zero and 255. This means each digital bit equals 19.5 millivolts. The syntax is READADC, I/O port (with the attached photometer), byte-sized RAM variable (to store the result). So, for example, **READADC C.0,B0** digitizes the voltage on I/O port C.0 and stores the result of the conversion into RAM variable B0.

The second PICAXE ADC command coverts a zero to five volt sensor output into a 10-bit value between zero and 1,023. This means each digital bit equals 4.8 millivolts. The syntax is READADC, I/O port (with the attached photometer), word-sized RAM variable (to store the result). Again, for example, **READADC C.0,W0** digitizes the voltage on I/O port C.0 and stores the result of the conversion into RAM variable W0.

If the digitized value will not exceed 255, then the result of the 10-bit conversion can be stored in a byte-sized variable. However, this will limit the range of sensor voltages from zero volts to 1.25 volts.

Which command to use is up to you. Remember, however, that the 10-bit conversion requires one word (two bytes) of memory to store (as opposed to only one byte of memory to store the eight-bit conversion). However, the resolution afforded by the 10-bit conversion is four times greater than the resolution of the eight-bit conversion.

Now that we have a way to record the voltage from the LED photometer, let's begin the calibration process. The output of the LM335 temperature sensor only changes because of the temperature. As the temperature increases, so does the output voltage of the LM335.

The current from the LED — and therefore the voltage from the LED — photometer changes due to two factors: the light intensity and the temperature.

Therefore, as the light intensity increases, so does the voltage output of the transimpedance amplifier. Unfortunately, the same is true for temperature. So, we have the situation where even while the light intensity remains constant, the output from the photometer will change if the temperature of the photometer changes.

The way we will get around this problem is by calibrating the photometer's temperature response.

Calibrating The LED Photometer Output

Calibrating the LED photometer's temperature response requires monitoring the output from the circuit while its temperature changes. The easiest way to accomplish this is to chill the photometer circuit and then measure both the temperature and the LED outputs while the light intensity remains constant and its temperature increases.

Because the photometer is used to measure the intensity of sunlight, this test should be performed outside on a sunny day.

Begin programming a flight computer to record the photometer's temperature and light intensity. Data should be recorded at least once per second, and don't forget that the flight computer's program must also download the data when the data collection is complete.

This last point is critical if you are storing the data inside the PICAXE's EEPROM. A PICAXE cannot be reprogrammed to download data without wiping out the data already stored on the chip.

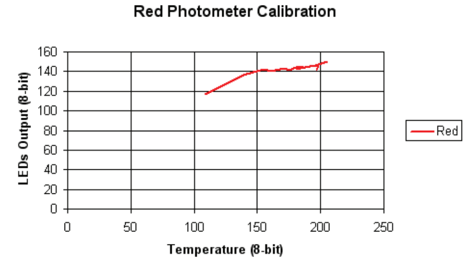
Next, find a sunny location that you can reach quickly from the kitchen freezer. At that location, you must hold the photometer absolutely still because any movement of the photometer will change the intensity of sunlight shining on the LED, potentially adding variation to the calibration data.

Finally, the calibration process needs to take place at a time when condensation from the atmosphere will not cover the LED. So, wait for a relatively dry day for this activity.

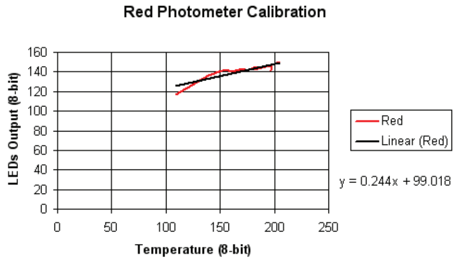
Now, chill the photometer in the freezer. Make sure frost doesn't form over the LED while it is chilling.

109	117
139	136
152	141
158	141
160	140
160	140
164	141
172	143
175	141
181	144
186	145
181	142
188	144
196	146
205	150
196	146
198	145
197	141

■ **TABLE 1.** A sample of data collected by a red LED-based photometer. The temperature reading is the first column of data, and the second column is the photometer reading. Both readings are digitized with eight-bit resolution.



■ **FIGURE 1.** This is the chart that was created from a red LED-based photometer. You can see that I made a note in the chart that both axes are from eight-bit data.



■ **FIGURE 2.** Excel generated this trendline and equation, based on the data from Figure 1.

After the photometer is chilled, carry it quickly to a sunny location and begin recording voltages. The photometer should reach outdoor temperature after a few minutes. Bring the photometer back inside and download the data.

I suggest using the Terminal program in the PICAXE Editor for retrieving the calibration data. After copying and saving the data in a text file, import it into a spreadsheet. The data will look like what you see in **Table 1**.

Now, create a graph based on the data. For the axes, make temperature the x-axis (or independent data) since temperature was the variable that we changed. Make photometer output the y-axis (or the dependent data) since the photometer output changed in response to the changing temperature. The resulting chart should look like **Figure 1**.

If using Excel, you can elect to create a trendline for the calibration data. If your spreadsheet does not offer this option, then draw the best fitting straight line through the calibration chart and generate the equation describing the slope and intercept of the line. If using Excel, select a linear trendline.

Also, select to include the equation for the trendline into the chart. The result will look like what you see in **Figure 2**.

For my calibration data here, temperature affects the photometer's output according to the equation, $y = 0.244x + 99.018$.

In this equation, *y* is the output of the LED and *x* is the photometer's temperature.

Once you have the photometer's calibration equation, it will be used to correct light intensity data.

Using The LED Photometer

Step 1

Collect your data during a near space mission and download the results through the PICAXE Terminal. Then, copy the data and paste it into a text file. Import the text file into a spreadsheet as two columns of data: photometer temperature and light intensity.

Step 2

Use the spreadsheet to calculate the photometer's expected output based on the temperature of the photometer. The result is corrected for temperature, and assumes that the light intensity didn't change from the time the calibration data was collected.

Step 3

Divide the LED's actual output by the expected output. This ratio is the photometer's corrected output during the flight, relative to the calibration reading.

Step 4

Now, compare each corrected output during the flight to the initial corrected output on the ground. Do this by dividing each corrected output by the initial corrected output on the ground. The result shows how the intensity of sunlight varied during the flight.

This data should be graphed in relationship to the altitudes at which the data was collected.

In most cases, the independent variable (altitude in this case) is plotted along the x-axis. However, the graph is more informative if the altitude is plotted along the y-axis.

An Issue To Resolve

The LED in the photometer will never be insensitive to its pointing direction. Therefore, you will find that the intensity data varies as the BalloonSat rotates in relationship to the sun.

Here are a few ideas for addressing this issue.

First, you can combine the LED photometer with a sun sensor. Then, the flight computer could only collect data when the photometer is properly oriented relative to the sun.

Second, you can cover the LED photometer with a diffuser — like half a ping pong ball. A diffuser will make the sky appear more uniformly illuminated to the LED and eliminate some of the photometer's pointing sensitivity.

Third, you can program the flight computer to collect data very rapidly. The data would still show a cycle of the photometer fluctuations as it rotated into and out of the sun.

However, you can select the highest value of each cycle and discard the rest of the data. What data remains will display less variation from the photometer's rotation.

Forth, a camera could be incorporated into the photometer experiment. The camera could photograph the pointing direction (relative to the sun) of the photometer each time a photometer reading is recorded.

Alternately, the camera could record the angle of a shadow cast by a dowel at each photometer reading. In either case, only the data collected while the sun was in the proper location would be kept and the rest discarded.

Then, a comparison of the data will show less variation due to the photometer rotating.

Fifth, use several LEDs as input to the TLC272 and point each one in a different direction to get more uniform coverage of the sky.

That does it for the LED photometer. If you get a chance to try out this experiment, please let me know your results. I would especially be interested in results comparing two different colors of LEDs simultaneously.

Onwards and Upwards,
Your near space guide **NV**

Converting Temperature Signals Into Temperatures

Since the photometer needs temperature data, it would be a shame not to use it to measure the temperature of the atmosphere. In the LM335 temperature sensor, one volt represents 100 Kelvins. The Kelvin scale is the Celsius scale adjusted for Absolute Zero.

Use the following mathematical procedure to convert the result of the LM335's ADC reading into the temperature in Kelvins.

Eight-bit READADC Reading

Temperature (K) = (ADC Value / 256) X 500

Ten-bit READADC10 Reading

Temperature (K) = (ADC Value / 1024) X 500

Then, convert the units of Kelvins into units of Celsius by subtracting 273.

ºC = K - 273

The Celsius scale can be converted to the Fahrenheit scale two ways. A really slick conversion method takes advantage of the fact that there is 1.8 Fahrenheit degrees for every 1.0 Celsius degrees, and that the Celsius and Fahrenheit scales intersect at -40 degrees. With this in mind, the conversion is as follows.

ºF = ((ºC + 40) * 1.8) - 40

ALL ELECTRONICS CORPORATION

QUALITY Parts
FAST Shipping
DISCOUNT Pricing

CALL, WRITE, FAX or E-MAIL for a FREE 96 page catalog.
Outside the U.S.A. send $3.00 postage.

HIGH-TORQUE SERVO

Z590M racing servo. Metal gears. 85 oz-in torque @ 6Vdc. Speed: 0.18 sec/60° @ 4.8V, 0.15 sec/60° @ 6V. Standard body size: 1.50" x 0.74" x 1.37" high. 7" leads with 3-pin JR connector.

CAT# DCS-107

5 for $10.65 each

$10.95 each

RETRO-FIT 2.1MM COAX SOCKET

An in-line socket with screw terminals that mates with 2.1mm coax power plugs. An easy, no-solder way to connect LED strips and lamps, video cameras and other devices to a power source with standard 2.1mm barrel connectors.

CAT# SDP-S

$2.20 each

7' POWER CORD W/ HOSPITAL GRADE PLUG

Hospital grade, green dot, 3-prong plug. Blunt cut-off other end. Rated 10A 125V. 18/3, SJT, 105C, 300V cable. CSA. Removed from new equipment, excellent condition. **CAT# LCAC-669**

100 for $2.00 each

$2.50 each

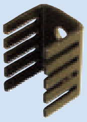
SPRING TERMINAL STRIPS

Quick-release spring-loaded terminals. Accept up to 12AWG wire. Red/black color-coded levers. Solder-loop terminals.

2-terminal
CAT# SLT-25
2 for $1.00
100 for 35¢ each

4-terminal
CAT# SLT-45
95¢ each
10 for 80¢ each • 100 for 70¢ each

TO-220 HEATSINK



Wakefield # 274-2. Black anodized aluminum heatsink for TO-220 devices. 19.1 x 13.2 x 12.7mm.

CAT# HS-333

100 for 20¢ each

4 for $1.00

6-12 VDC MOTOR

Nichibo # UC-280P-19160TBNRCE
No-load specs: 11,500 RPM @ 12.0 Vdc, 0.15 A. At maximum efficiency: 9,000 RPM @ 12Vdc, 0.57A - 46 g-cm torque. 30.5mm long x 18.3mm x 24mm. 2mm dia. x 11mm long shaft. Motor prepped with three 0.1 uF capacitors across power terminals to ground.

CAT# DCM-427

10 for 90¢ each • 200 for 80¢ each

$1.00 each

MOTOROLA "HANDS FREE" HEADSET

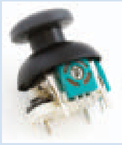
This comfortable, hands-free headset provides great sound quality on both ends of the conversation. The earpiece lays on the ear, outside of the ear canal, eliminating the discomfort associated with in-ear headsets. 2.5mm stereo phone plug. The 2.5mm plug is standard on most cordless home phones. However, most current cell phones / smart phones require a 3.5mm plug. Use adapter CAT# AD-25.

CAT# PHN-39

10 for $2.00 each

$2.35 each

THUMB JOYSTICK W/ SELECT BUTTON



2-Axis, self-centering Joystick contains two independent 10K Ohm pots (one per axis) for reporting the joystick's position. Select button actuated when the joystick is pressed down.

CAT# JS-932

10 for $3.56 each

$3.95 each

BULKHEAD MOUNT JUNCTION BOX

2.54" square x 1" junction box. Black, break-resistant ABS plastic. Tight-fitting, lap-joint lid with counter-sunk screws. Mounting flanges with holes on 3" centers. There is a 0.5" diameter hole one side, and a 1.5" x 0.25" slot in the rear of the box.

CAT# MBF-30

10 for $1.50 each

$1.65 each

SOLAR CELL, 60MM X 60MM X2MM

Output: approximately 3 Volts @ 40 mA. 60mm square x 2.5mm thick epoxy-encapsulated silicon photo-voltaic cell. Solid, almost-unbreakable module with solderable foil strips on backside. Ideal for solar-powered battery chargers and other projects.

CAT# SPL-61

100 for $3.25 each

$3.75 each

24 VDC 0.65A POWER SUPPLY

SMP Model SAD224. Input: 100-120V 50/60Hz 0.4A. Output: 24Vdc 0.65A. 6' cord. 2.1mm i.d. coax power plug, right-angle. Center positive.

CAT# PS-2406

10 for $7.50 each

$7.95 each

33,000UF 50V COMPUTER GRADE CAPACITOR

Nippon Chemi-con U32D 105C. 35mm dia. x 123mm high (excluding terminals). Screw terminals.

CAT# EC-335

10 for $4.50 each

$4.75 each

Shop ON-LINE www.allelectronics.com
ORDER TOLL FREE 1-800-826-5432

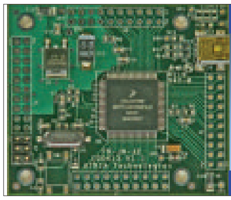
MAIL ORDERS TO:
ALL ELECTRONICS CORP.
14928 OXNARD ST., VAN NUYS, CA 91411-2610

FAX (818) 781-2653 • INFO (818) 904-0524
E-MAIL allcorp@allcorp.com

NO MINIMUM ORDER • All Orders Can Be Charged to Visa, Mastercard, American Express or Discover • Checks and Money Orders Accepted by Mail • Orders Delivered in the State of California must include California State Sales Tax • NO C.O.D • Shipping and Handling $7.00 for the 48 Continental United States - ALL OTHERS including Alaska, Hawaii, P.R. and Canada Must Pay Full Shipping • Quantities Limited • Prices Subject to change without notice.

MANUFACTURERS - We Purchase EXCESS INVENTORIES... Call, Write, E-MAIL or Fax YOUR LIST.

ELECTRONET



CanaKit™

Electronic Kits • Modules • Parts

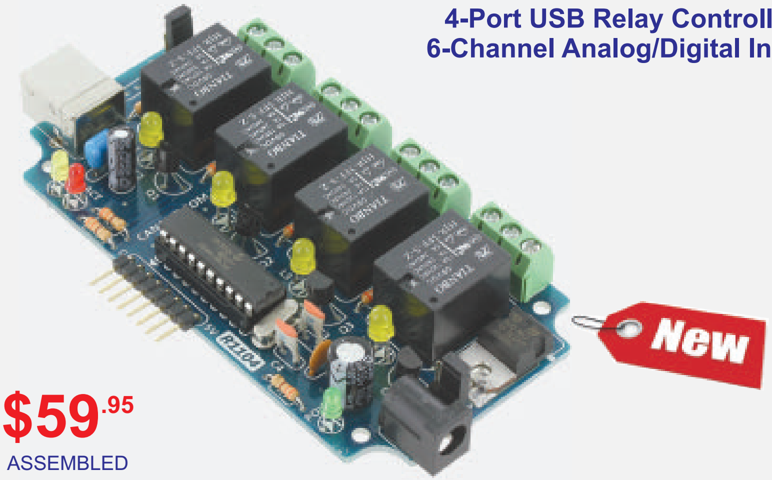
Save 10% using Coupon Code NV2012*
when ordering online at www.canakit.com

Motor Controllers • Arduino • Signal Generators • FM Transmitters • Timers • Audio Amplifiers

200 KHz Function Generator

$49.95
ASSEMBLED
UK307

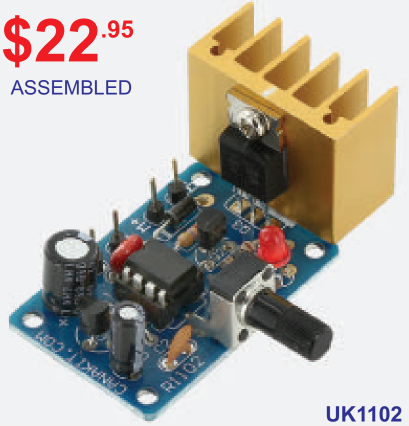
4-Port USB Relay Controller with 6-Channel Analog/Digital Interface



$59.95
ASSEMBLED
UK1104

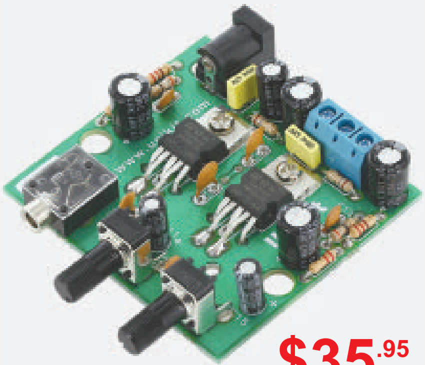
5A Motor Speed Controller (PWM)

$22.95
ASSEMBLED



UK1102

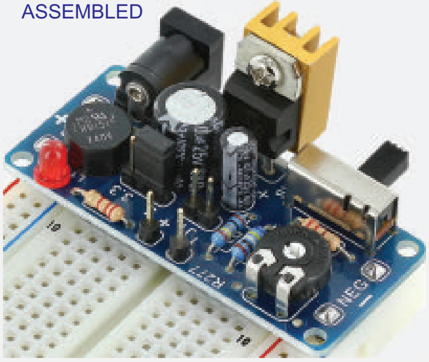
Stereo Audio Amplifier (10W)



UK560
$35.95
ASSEMBLED

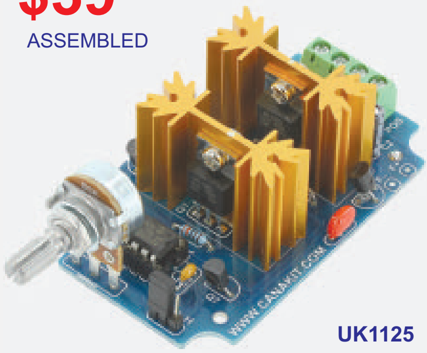
Plug-In Breadboard Power Supply

$19.95
UK277
ASSEMBLED



7A Bi-Directional Motor Speed Controller (PWM)

$39.95
ASSEMBLED



UK1125

USB PIC Programmer (MPLAB Compatible)

$69.95
ASSEMBLED
UK1300

Digital LED Timer (Up/Down)

UK258

$46.95
ASSEMBLED

1•888•540-KITS (5487)   info@canakit.com

www.canakit.com

* Limit one per customer. Can not be combined with other coupons or volume discounts.
Does not apply to products already on sale. Prices subject to change without notice.

BY JOE PARDUE

Arduino Handheld Prototyper

Part 1

Go to **www.nutsvolts.com/index.php?/magazine/article/july2013_SmileysWorkshop** for any additional files and/or downloads associated with this article. You can also discuss this topic at **http://forum.nutsvolts.com**.

Last month, we finished up our extended series on Arduino/Fritzing prototyping to production. You should now be able to create excellent documentation for your designs that includes images for your breadboards, schematics, and layouts for printed circuit boards (PCBs). These skills will come in very handy as you proceed into the world of Arduino-based systems development and beyond. This month, we are going to start a two-part series on an Arduino-based handheld prototyper shown in Figure 1. We will have Fritzing components for this system (Figure 2), so we can continue using our excellent Fritzing breadboard and schematic images to document our work as before. The Arduino handheld prototyper — as the name implies — let's you develop prototypes that are portable in your very own hand, plus provides a bonus that is lacking in the stand-alone Arduino: It can accept user keypad input via pushbuttons and provides the user with visual output via an LCD.

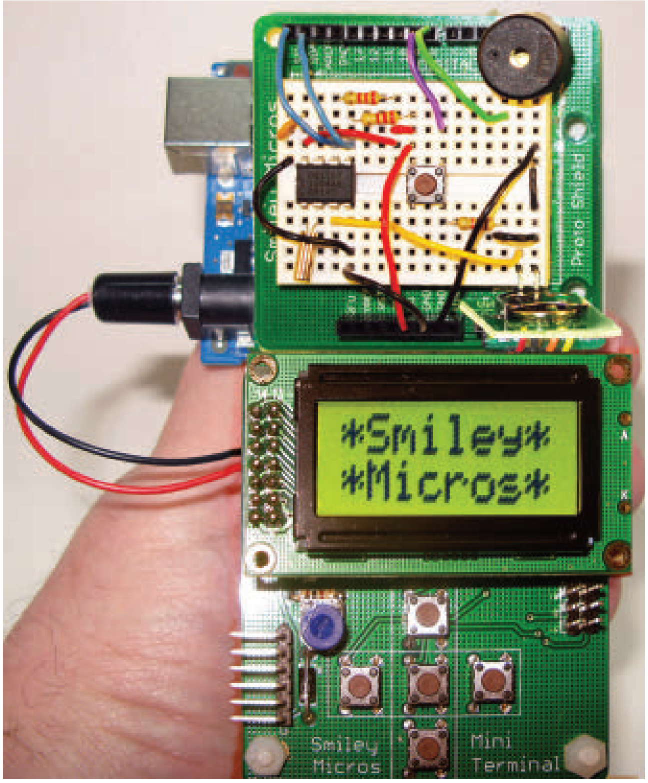


FIGURE 1. Arduino handheld prototyper.

Excelsior!

In case you're wondering, the title to this section means either 'still higher' or 'fine curled wood shavings used to pack fragile items.' My use is in the 'still higher' sense of things, meaning we've just finished a steep climb developing our prototyping skills with Fritzing and the Arduino proto shield. Now, we are going still higher.

Think of this as mountain climbing. You've just arrived at the top of the foothills next to the base of the mountain. You've got a long way to go, but you've put in quite an effort to get this far. Time to stop and look around, to gaze off in the distance, and rest a bit before the next climb. I've been climbing a long time, and one thing I've noticed is that each time I reach a peak, I look up and see a higher peak. The learning never ends, but that is a big part of its attraction. Excelsior indeed!

For our next climb, we are going to learn to use the handheld prototyping system we'll be discussing here so that we can continue our Arduino development without having to be tethered to a PC all the time. We'll take all the stuff we've recently learned and combine it with what we covered about the LCD Navigator way back in the January and February 2012 Workshops.

The Arduino Handheld Prototyper

Our exercises so far have almost all been tethered to a PC via a USB cable. That is really great for learning because you have a robust power supply built in, and you have immediate access to all the great development and debugging tools available on a PC. Microcontroller designs are often used away from a PC, and are embedded deeply into other systems (for example, controlling the sensors and actuators in an automobile engine) or are carried around by the user as handheld instruments like a multimeter. In fact, being away from big computers is one of the main reasons that folks use microcontrollers. Generally, at some point, for an embedded system to be really embedded, it must get away from a PC. So, we need a way to develop for non-PC-tethered applications.

The Arduino handheld prototyper is made from three separate subsystems. You may recognize the upper part from **Figures 1** and **2** as the Arduino proto shield we've been using for the past few articles. The lower PCB looks a lot like that LCD Navigator mentioned above — but it isn't. This one has brains.

The LCD Navigator had a parallel interface and required a microcontroller to run it. However, the I^{2}C mini terminal has a built-in microcontroller (ATmega328P) that operates the LCD, reads the pushbuttons, and communicates via the I^{2}C bus.

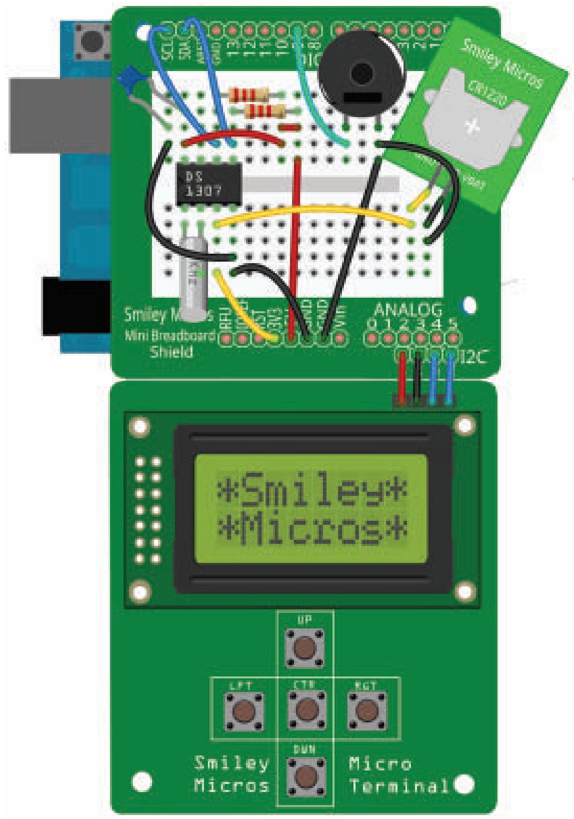


FIGURE 2. Arduino handheld protoyper in Fritzing.

Introducing The Serial Mini Terminal

The LCD Navigator is great for prototyping LCDs and buttons on a breadboard, but it uses up too many of the Arduino pins to be practical for a handheld device. The I^{2}C mini terminal only uses the two I^{2}C lines from the Arduino, leaving all the rest of the I/O pins available for prototyping on the proto shield.

I took the LCD Navigator design and ported it to our Fritzingduino concept (meaning that I added a minimal Arduino clone to it). **Figure 3** shows the I^{2}C mini terminal with the LCD removed. You can see that under the LCD is a circuit that looks an awful lot like the Fritzingduino. It has an ATmega328P with an Arduino-compatible bootloader, along with associated crystal, reset, and communication circuitry. You can communicate with the ATmega328P using the FTDI connector as discussed for the Fritzingduino, and/or you can program it with the ISP header.

It is a fully functional (minimal) Arduino clone; it has a bootloader and can communicate with a PC, and it is pre-loaded with the I^{2}C mini terminal application so that it can function as a terminal server for the Arduino under the proto shield. In its intended use, you'll never have to utilize the bootloader and change the application. However, if you get a clever idea that you want to test with this board using it as an Arduino clone but with user input and output, there is nothing to stop you! Hack away and be sure and tell me if you make it better. [Note that the IC in **Figure 3** has two yellow dots on it. These aren't eyes; they are a marking that indicates this ATmega328P has an Arduino-compatible bootloader, the mini terminal software, and that it has been tested.]

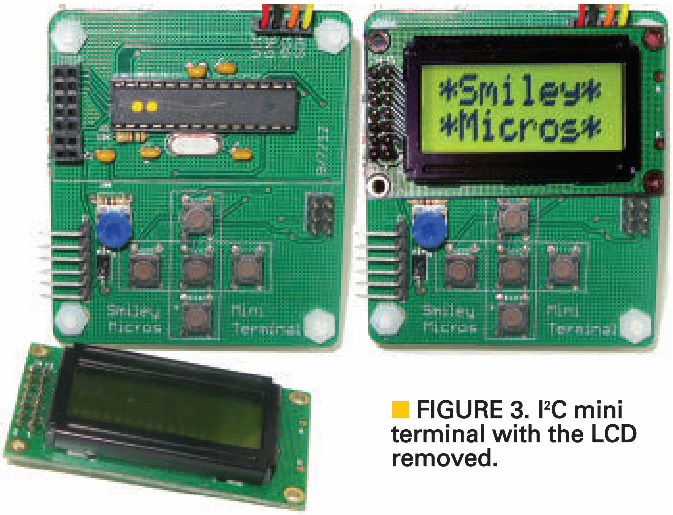


FIGURE 3. I^{2}C mini terminal with the LCD removed.

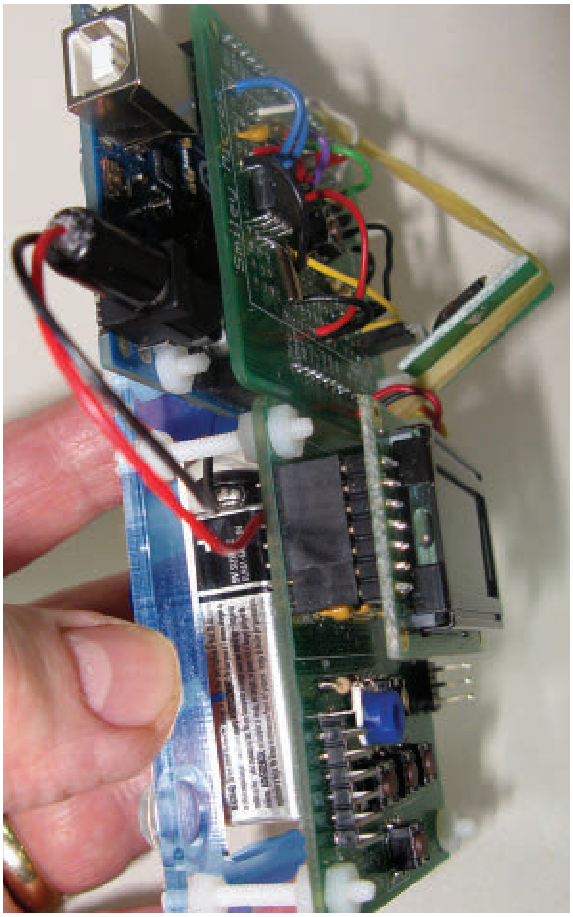


FIGURE 4. Side view showing the base with a slight blue tone.

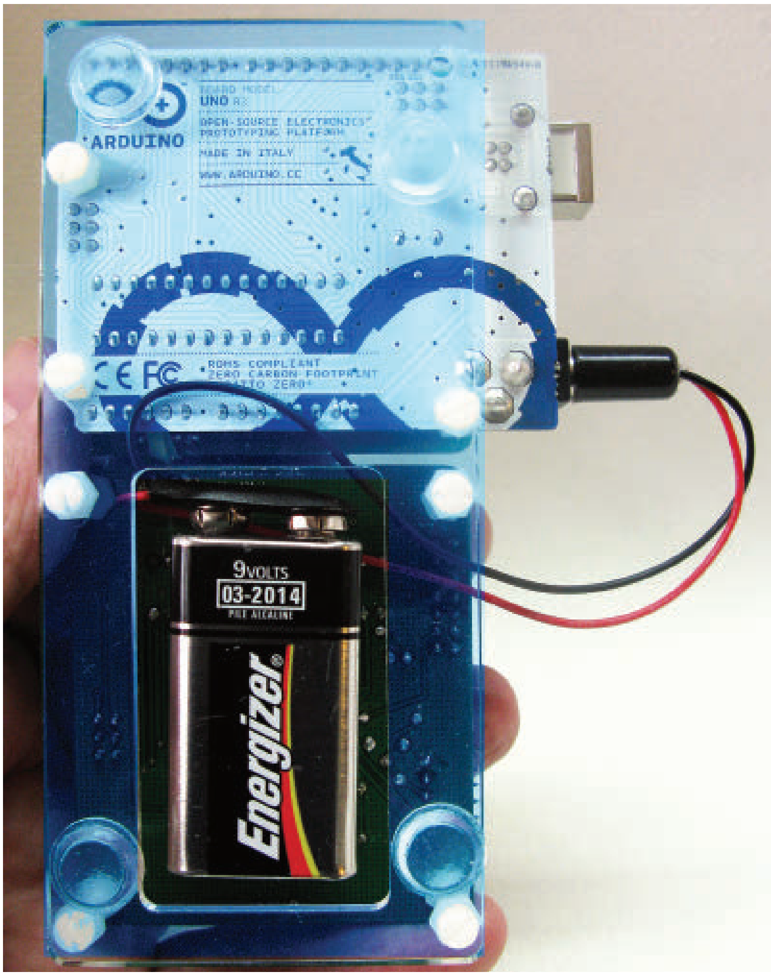


FIGURE 5. Back view.

This design concept means that you are actually using two Arduino-compliant ATmega328P microcontrollers in one handheld: the ATmega328P on the mini terminal and the ATmega328P on the 'real' Arduino under the proto shield. The mini terminal ATtmega328P behaves as an I^{2}C slave to the proto shield I^{2}C master, puts characters on the LCD when told to do so, and returns the status of the buttons when instructed. We will look in detail at the slave software next month.

The really great thing about using I^{2}C on the Arduino is that it has a novice-friendly library built into the Arduino IDE, so you can develop handheld systems that use the LCD and the buttons without having to understand anything about I^{2}C. You use it as a simple black box and leave the details to the experts.

The Base

To make this development tool 'handheld,' I've designed a plastic base shown in **Figures 4** and **5** that you can bolt the Arduino proto shield and mini terminal to. It even has a cutout so that you can stick a nine volt battery to the back of the mini terminal with Velcro™ and be able to change the battery easily. The base is laser-cut from clear plastic and is sort of invisible in a photograph, so for **Figures 4** and **5**, I Photoshopped a blue tone so that you can see it better. The whole thing is held together with nylon nuts and bolts, making it easy to assemble and disassemble

I would expect a user of this system to develop a somewhat Frankenstein looking handheld tool that works great even though it looks like crap [note the rubberband I neglected to hide in **Figure 4**]. Then, once it is perfected, you can put on some makeup and lipstick. Actually, the point is that after you get it all working, then you can decide how you need to package your creation so that Elsa Lanchester won't run off screaming when you put you project on Kickstarter. [That metaphor should win an award].

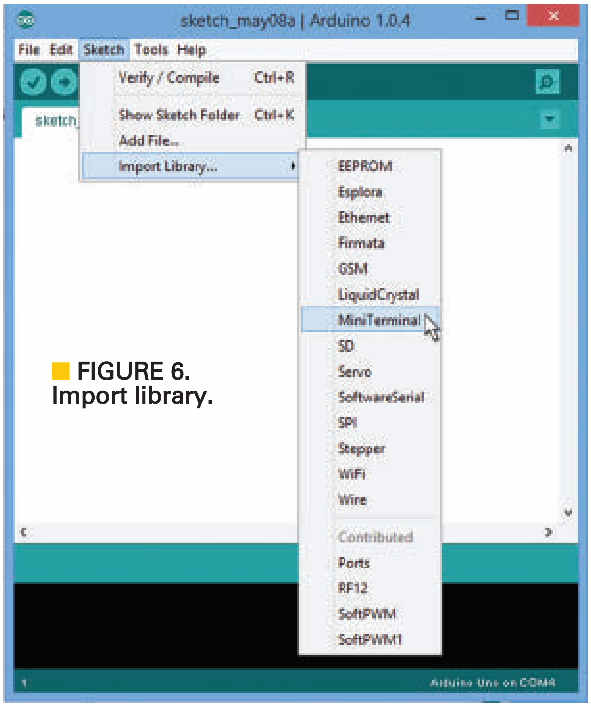
Building The Serial Mini Terminal

Rather than spend a lot of time and space here showing step-by-step how to build the I^{2}C mini terminal, I'll just refer you to the Smiley Micros blog entry with the note that if you DIY, you'll need that ATmega328P to have an Arduino-compatible bootloader on it. [In case you're interested, there is an I^{2}C mini terminal kit available from the *Nuts & Volts* webstore.]

In last month's Workshop, I discussed how to put a bootloader on an ATmega328P using an Arduino as an ISP programmer. What I didn't mention is that everybody seems to call the ATmega328P the ATmega328, as I did. Well, they are wrong just like I was.

There is an ATmega328 and an ATmega328P, and they are different beasts. It is not just that the 'P' model uses less power, it also has a different signature. So, when you try to use the ArduinoISP program which calls avrdude, the signature gets checked and it won't like it if the signature doesn't match. Even the Arduino documentation for the ArduinoISP software just calls the chip an ATmega328 — IT IS WRONG!

I know this because I have both chips and I could only get the ArduinoISP to work with an ATmega328P. You can get it to use the ATmega328, but you've got to jump through so many hoops that is just isn't worth it IMHO. Stick with the ATmega328P and save some time and trouble.



■ FIGURE 6. Import library.

How To Use The Arduino Handheld Prototyper

Let's look at a little program that has the Arduino in the proto shield read a button on the mini terminal and then tell the LCD display on the terminal to show the name of the button. This not only demonstrates the basic functions of the mini terminal, it shows how easy it is to use it. Easy is good. First, however, a word or two that might help prevent confusion.

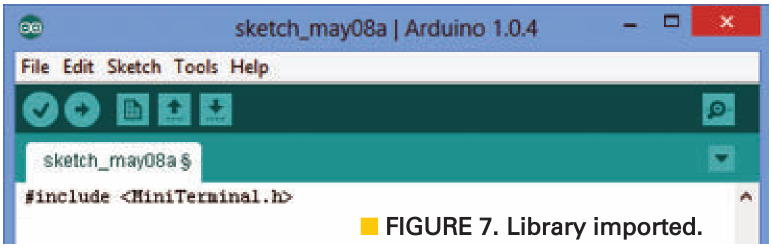
We are going to use two sets of code. One set is resident on the I^{2}C mini terminal and operates the LCD and the pushbuttons — this is the slave code. The other set of code runs on the Arduino and is the master code. A lot of the master code puts stuff on the LCD, but it does this by telling the slave code to do it. It does the same for reading the pushbuttons.

You will need to install the mini terminal library in your Arduino IDE. You can get this library and testing software in a single file called MiniTerminal.zip in the article downloads. Unzip this file and then copy the MiniTerminal folder to the libraries section of your Arduino installation. In my case, this was C:\Arduino-1.0.4\libraries\Mini Terminal.

To use this library, you open your Arduino IDE and (as shown in **Figure 6**), open the Sketch menu and select Import Library...\MiniTerminal. This will write to the top of the program the *#include <MiniTerminal.h>* required to use the library as shown in **Figure 7**.

The following *mini_terminal_master* software runs on the Arduino:

```
// mini_terminal_master Joe Pardue 5/9/13

#include <MiniTerminal.h>
#include <Wire.h>

// Store strings in program memory
p_string(Hello) = "Hello";
p_string(world) = "world!";
p_string(up) = "up";
p_string(down) = "down";
p_string(center) = "center";
p_string(left) = "left";
p_string(right) = "right";
p_string(error) = "error";

int oldMillis = 0; // used in checking for
button presses
byte last_button = 0;

void setup() // one time functions to get
started
{
  Wire.begin(); // join i2c bus (address
  optional for master)

  mt_home(); // Tell the LCD to put the cursor
  an the upper left
```



```
#include <MiniTerminal.h>
```

■ FIGURE 7. Library imported.

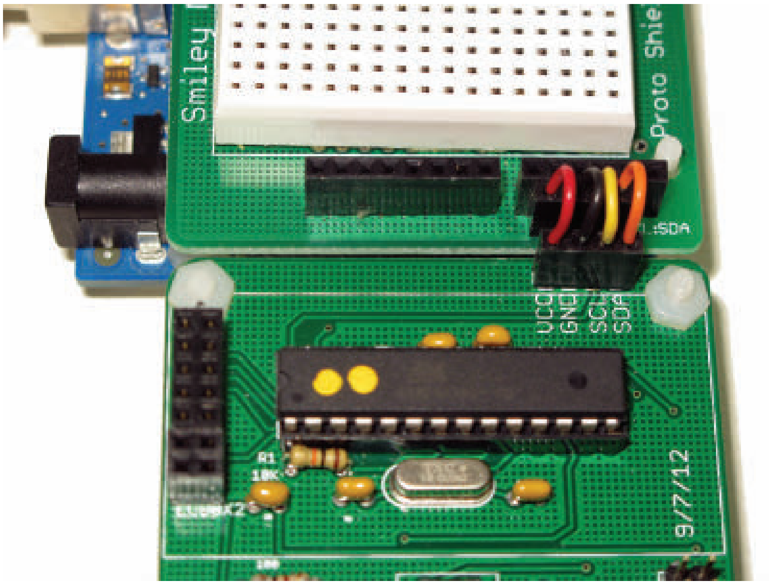


FIGURE 8. Wire the I²C and power.

```
  mt_clear(); // Clear the LCD
  mt_print(Hello,0,0); // Print Hello on line 1
  mt_print(world,0,1); // Print world! on line 2
}

void loop() // repeat functions forever
{
  // use millis() to check for available buttons
  // about 10 times a second
     if( (millis() - oldMillis) > 100)
     {
       last_button = mt_get_button();
       oldMillis = millis();
     }

     // see if a new button has been pressed
     if(last_button){
       show_button();
       last_button = 0;
     }
   }
   // Send the last button requested back to
   // the LCD to display
   void show_button()
   {
     mt_clear();

     if(last_button == LFT)
     {
       mt_print(left,0,0);
     }
     if(last_button == UP)
     {
       mt_print(up,0,0);
     }
     if(last_button == CTR)
     {
       mt_print(center,0,0);
     }
if(last_button == RGT)
{
  mt_print(right,0,0);
}
if(last_button == DWN)
{
  mt_print(down,0,0);
}
if(last_button == 0)
     {
       mt_print(error,0,0);
     }
   }
```

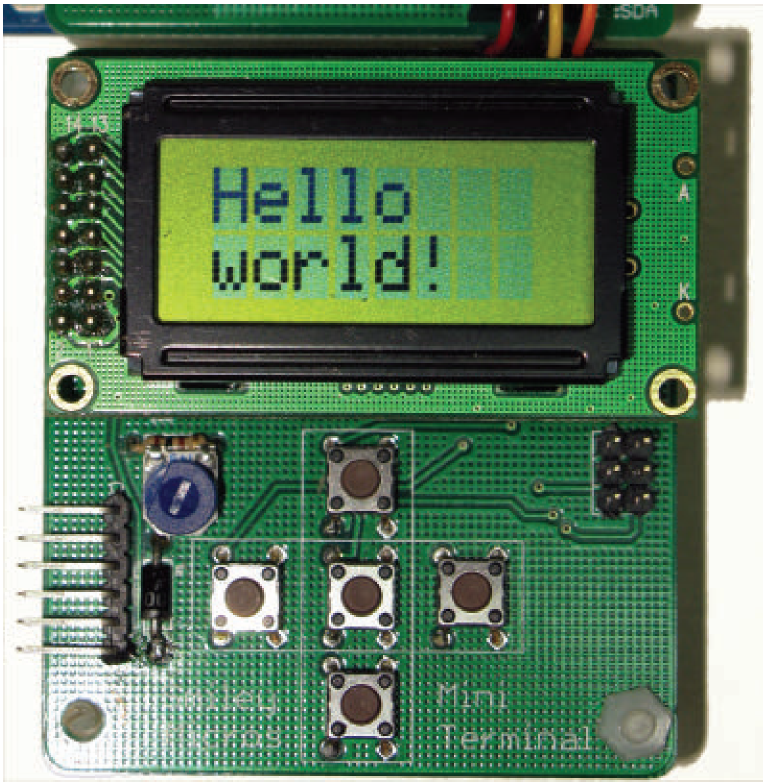


FIGURE 9. Say Hello World!

Save Some SRAM

The first thing you may notice is the section beginning:

```
// Store strings in program memory
p_string(Hello) = "Hello";
…
```

This uses a *macro p_string* located in the *MiniTerminal.h* header file that lets us store strings of data in program memory. In microcontrollers, SRAM that is used to run the program tends to be precious and small, while the program memory tends to be stored in a cheaper and larger type of memory such as data Flash in the AVR. However, the C (and C++) compilers gcc and g++ (used by the Arduino) will put character strings in SRAM.

The version of these compilers used by the Arduino has some special features that allow you to store and retrieve strings in the cheaper program memory, but these features are somewhat arcane and frankly, a bit much to throw at typical Arduino users. I won't burden you at this time by showing you the macro or the functions that use the strings; you can look at it in the header file if you are curious.

It might seem like a bit of overkill to go to all that trouble since (in this example) we are only using a few short words, but in a real world application you might want to present a lot of information to the user. So, it is good to have an option that won't run you out of memory.

Press Some Buttons

To run the mini terminal from an Arduino, you need to connect the +5, GND, and the two I^{2}C lines SCL and SDA on to the Arduino proto shield as shown in **Figure 8**. (You can also hook these wires directly to the Arduino if you don't have the shield. Just make sure you get the SCL and SDA wires to the correct socket position.) Next, you load the *mini_terminal_master* program shown above into the Arduino. You should see the 'Hello world!' message shown in **Figure 9**.

Now, start pressing the buttons. The button presses are read by the Arduino and then the name of the button pressed is sent back to the LCD as shown in **Figure 10**. When you use this later for a real world application, the up, down, left, and right buttons will be used to scroll around in menus. We will use the center button shown in **Figure 11** to select a menu item.

You are probably already familiar with the paradigm since it is pretty much what all TV remote controls do with their similarly arranged five pushbutton keypad. Of course, they generally have a lot more keys for things like selecting the number of a channel, but as you'll see next month, you can do a lot with these simple keys including inputting numbers without having special number keys.

Well, that should get you started. Next month, we'll look a little deeper at the master software and then look at the slave code. We will then apply the Arduino handheld prototyper to a practical project that helps heat and cool a castle by controlling the fresh airflow based on reading indoor and outdoor temperature and humidity.

Did I say castle? Well, if you consider a large stone construction with a timbered Great Room, suits of armor, and broadswords a castle, then yes we are going to learn how one person uses our prototyper to help heat and cool his castle. (This is a perfect application for our new tool!) See you next month! **NV**

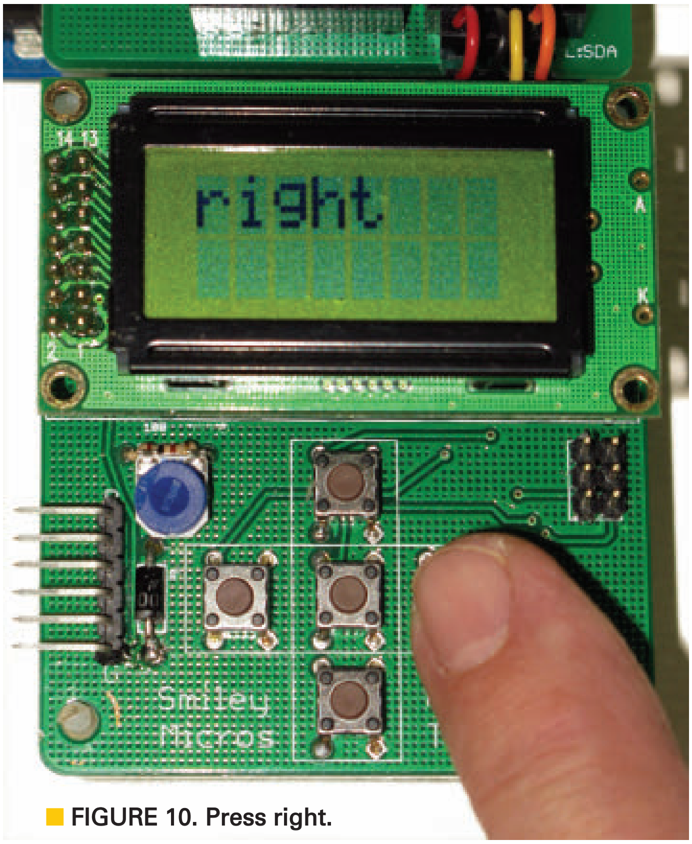


FIGURE 10. Press right.

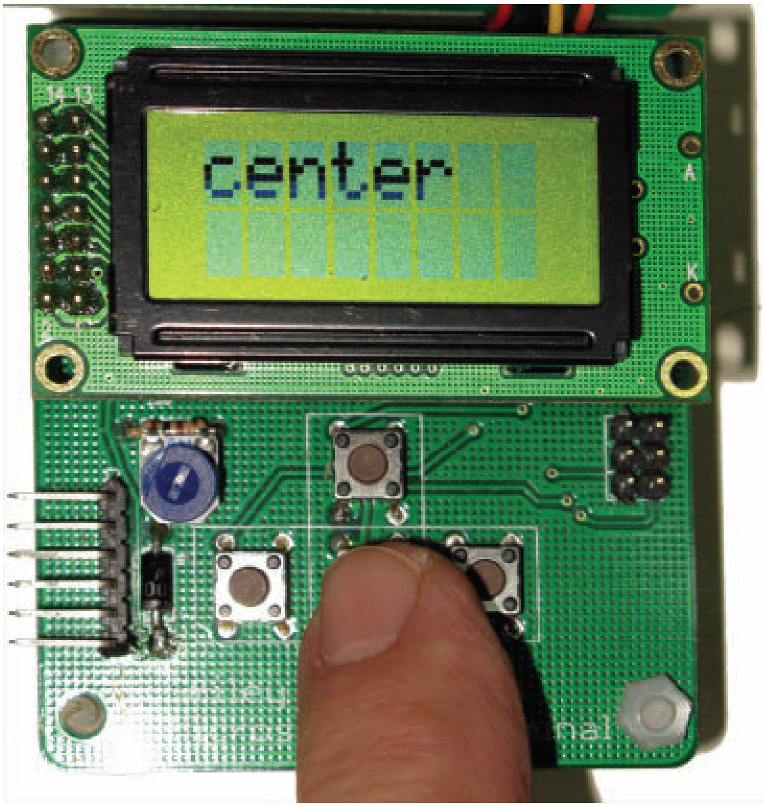


FIGURE 11. Press center.

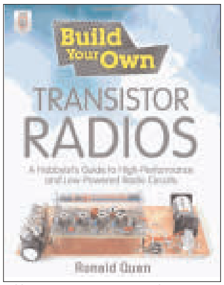
For complete product details, visit our webstore!!

The Nuts & Volts WEBSTORE

GREAT FOR DIYers!

Build Your Own Transistor Radios
by Ronald Quan

A Hobbyist's Guide to High Performance and Low-Powered Radio Circuits


Build Your Own
TRANSISTOR RADIOS
A Hobbyist's Guide to High-Performance and Low-Powered Radio Circuits
Ronald Quan

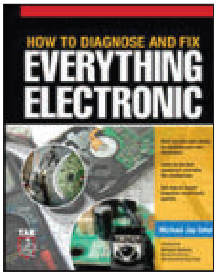
Create sophisticated transistor radios that are inexpensive yet highly efficient. Inside this book, it offers complete projects with detailed schematics and insights on how the radios were designed. Learn how to choose components, construct the different types of radios, and troubleshoot your work.

Paperback, 496 pages

$49.95

How to Diagnose and Fix Everything Electronic
by Michael Jay Geier

Master the Art of Electronics Repair!

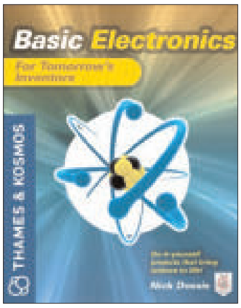


In this hands-on guide, a lifelong electronics repair guru shares his tested techniques and invaluable insights. *How to Diagnose and Fix Everything Electronic* shows you how to repair and extend the life of all kinds of solid-state devices, from modern digital gadgetry to cherished analog products of yesteryear. **About the Author:** Michael Jay Geier began operating a neighborhood electronics repair service at age eight that was profiled in *The Miami News*.

$24.95

Basic Electronics for Tomorrow's Inventors
by Nick Dossis

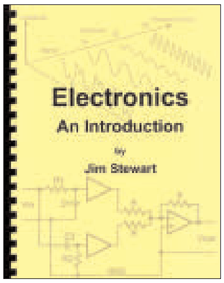
Learn all about electronics with hands-on experiments and projects.



This great book features a variety of simple, solder-free projects, including an LED reading light, electronic security keypad, IR target practice game, a real working telephone, temperature and moisture sensors, spy gadgets, and other neat stuff. Best of all, these experiments require only plug-and-play "breadboards" and other commonly available parts.

$19.95

Electronics An Introduction
by Jim Stewart

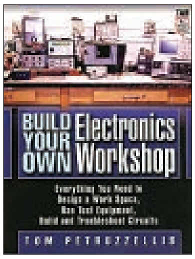


This book is designed as an in-depth introduction to important concepts in electronics. While electronics can be highly mathematical, this text is not about calculations. It is about how electronic equipment is able to extract, process, and present information held in electrical signals. If you are in — or studying to be in — a profession that requires the use of electronic equipment, then this book will provide the insight necessary to use such equipment effectively.

$33.95*

Build Your Own Electronics Workshop
by Thomas Petruzzellis

BUILD YOUR OWN DREAM ELECTRONICS LAB!


BUILD YOUR OWN Electronics Workshop
Everything You Need to Design a Work Space, Use Test Equipment, Build and Troubleshoot Circuits
TOM PETRUZZELLIS

This value-packed resource provides everything needed to put together a fully functioning home electronics workshop! From finding space to stocking it with components to putting the shop into action — building, testing, and troubleshooting systems. This great book has it all! And the best part is, it shows you how to build many pieces of equipment yourself and save money, big time!

Reg Price $29.95

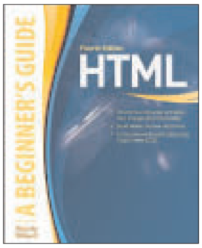
Sale Price $26.95

Beginner's Guide to ... Programming the PIC24/dsPIC33
by Thomas Kibalo

Kibalo takes you step by step through the fundamentals of programming the PIC24H which can equally be applied to the dsPIC33. His clear explanation of the inner workings make learning the PIC24H/dsPIC33 16-bit architecture easy. His code examples demonstrate how to perform the functions most applications require. The hardware is shown in a simple breadboard setup so even a beginner can build it, along with very few extra components needed.

$39.95*

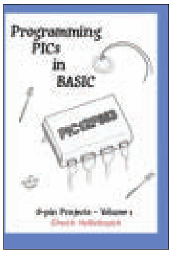
HTML: A Beginner's Guide
by Wendy Willard



Create highly functional, impressive websites in no time. Fully updated and revised, *HTML: A Beginner's Guide*, Fourth Edition explains how to structure a page, place images, format text, create links, add color, work with multimedia, and use forms. You'll also go beyond the basics and learn how to save your own web graphics, use Cascading Style Sheets (CSS), create dynamic web content with basic JavaScript, and upload your site to the web. By the end of the book, you'll be able to build custom websites using the latest HTML techniques.

$29.95

Programming PICs in Basic
by Chuck Hellebuyck



If you wanted to learn how to program microcontrollers, then you've found the right book! Microchip PIC microcontrollers are being designed into electronics throughout the world and none is more popular than the eight-pin version. Now the home hobbyist can create projects with these little microcontrollers using a low cost development tool called the CHIPAXE system and the Basic software language. Chuck Hellebuyck introduces how to use this development setup to build useful projects with an eight-pin PIC12F683 microcontroller. **$14.95**

Master and Command C for PIC MCUs
by Fred Eady

Master and Command C for PIC MCU, Volume 1 aims to help readers get the most out of the Custom Computer Services C compiler for PIC microcontrollers. The author describes some basic compiler operations that will help programmers particularly those new to the craft create solid code that lends itself to easy debugging and testing. As Eady notes in his preface, a single built-in CCS compiler call (output_bit) can serve as a basic aid to let programmers know about the "health" of their PIC code.

$14.95

Order online @ www.store.nutsvolts.com
Or CALL 1-800-783-4624 today!

BOOK & KIT COMBOS

16-Bit Complete Combo

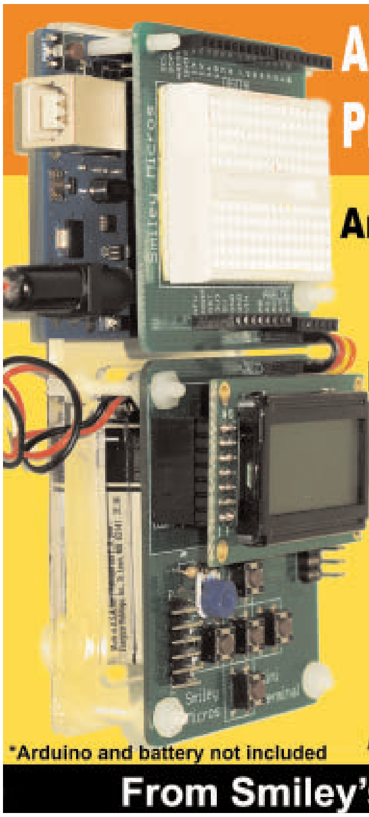
Combo Price $319.95

For complete details visit our webstore @ www.nutsvolts.com

Beginner's Guide Complete Combo!

Combo Price $229.95

For complete details, visit our webstore @ www.nutsvolts.com.



Arduino Handheld Prototyper System

Arduino Proto Shield $16.99

I2C Mini Terminal $29.99

Handheld Base $13.99

All three*: $59.99

*Arduino and battery not included

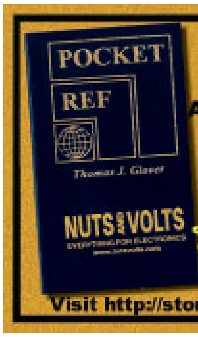
From Smiley's Workshop

CD-ROM SPECIAL

Nuts & Volts 9 CD-ROMs & Hat Special!

That's 108 issues. Complete with supporting code and media files.

Only $189.95 or $24.95 each.



The Nuts & Volts Pocket Ref

All the info you need at your fingertips!

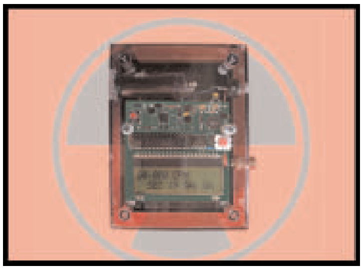
This great little book is a concise all-purpose reference featuring hundreds of tables, maps, formulas, constants & conversions. AND it still fits in your shirt pocket!

Visit http://store.nutsvolts.com or call (800) 783-4624

CALL 1-800-783-4624 today!
Order online @ www.nutsvolts.com

PROJECTS

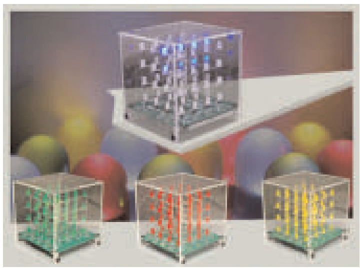
Geiger Counter Kit



This kit is a great project for high school and university students. The unit detects and displays levels of radiation, and can detect and display dosage levels as low as one micro-roentgen/hr. The LND712 tube in our kit is capable of measuring alpha, beta, and gamma particles.

Subscriber's Price **$145.95**
Non-Subscriber's Price **$148.95**

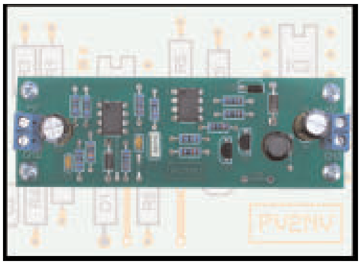
3D LED Cube Kit



This kit shows you how to build a really cool 3D cube with a 4 x 4 x 4 monochromatic LED matrix which has a total of 64 LEDs. The preprogrammed microcontroller that includes 29 patterns that will automatically play with a runtime of approximately 6-1/2 minutes. Colors available: Green, Red, Yellow & Blue

Subscriber's Price **$57.95**
Non-Subscriber's Price **$59.95**

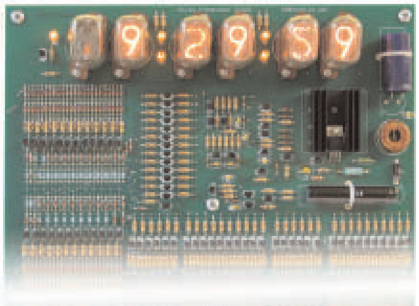
Inverter DC-DC Converter Kit



If you need +12V and -12V, but all that's available is +12V, then this project is for you. The inverted DC-DC converter gives you -V out when you put +V in, and works over a range of voltages without adjustment. It's an easy to build voltage mirror that can supply over 100 mA without a significant drop in voltage.

Subscriber's Price **$29.95**
Non-Subscriber's Price **$32.95**

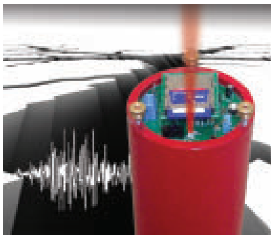
Neon Transistor Clock Kit



Add HIGH VOLTAGE to your clock!
This is a Nixie Tube display version of the Transistor Clock. It uses only discrete components — no integrated circuits. For more info, see the April 2012 issue.

Subscriber's Price **$245.95**
Non-Subscriber's Price **$249.95**

Seismograph Kit

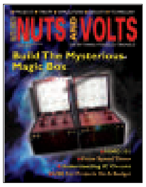


As seen in the May 2012 issue. Now you can record your own shaking, rattling, and rolling.

The Poor Man's Seismograph is a great project /device to record any movement in an area where you normally shouldn't have any. The kit includes everything needed to build the seismograph. All you need is your PC, SD card, and to download the free software to view the seismic event graph.

Subscriber's Price **$79.95**
Non-Subscriber's Price **$84.95**

Magic Box Kit

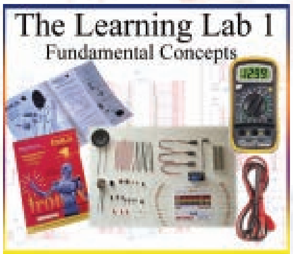


As seen on the April 2007 cover.

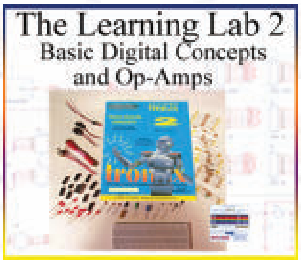
This unique DIY construction project blends electronics technology with carefully planned handcraftsmanship. This clever trick has the observer remove one of six pawns while you are out of the room and upon re-entering you indicate the missing pawn without ever opening the box.

Subscriber's Price **$39.95**
Non-Subscriber's Price **$45.95**

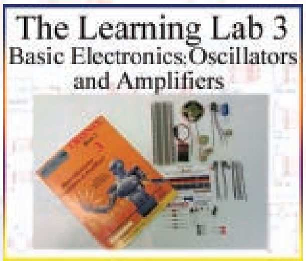
FOR BEGINNER GEEKS!



$59.95



$49.95



$39.95

The labs in this series — from GSS Tech Ed — show simple and interesting experiments and lessons, all done on a solderless circuit board.

As you do each experiment, you learn how basic components work in a circuit, and continue to build your arsenal of knowledge with each successive experiment.

For more info and a promotional video, please visit our webstore.

CLASSIFIEDS

KITS/PLANS

Electronic Kits

- Fun and Educational
- Discount White and Blue LEDs
- Quantity Discounts

www.bakatronics.com

ENGINEERING/ DESIGN SVCS

FLEXREV China Direct Manufacturing

Turn Key or Subassemblies

Components,Enclosures, Keypads

$0.007 per soldering point

2 layer PCBs $0.14/in2

Small jobs welcome

FLEXREV.COM

www.nutsvolts.com

ROBOTICS

Ultrasonic Distance Sensing Made EZ

HRUSB-MaxSonar®-EZ™
- Multi-Sensor operation
- USB interface
- Easy integration
- 1 mm resolution
- Calibrated beam pattern
- Starting at $49.95

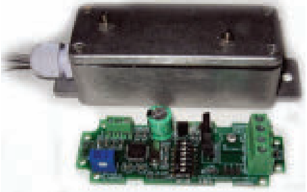
HRXL-MaxSonar®-WR™
- High noise tolerance
- IP67 rated
- 1 mm resolution
- Multi-Sensor operation
- Calibrated beam pattern
- Starting at $109.95

I2CXL-MaxSonar®-EZ™
- Great for UAV's and robotics
- Incredible noise immunity
- I2C interface
- 1cm resolution
- Automatic calibration
- Starting at $39.95

www.maxbotix.com

QKITS.COM

Mini DC Motor Controller



Up to 8A 35V; Adjustable acceleration time and frequency. Comes assembled in small metal case.

1-888-GO 4 KITS

COMPONENTS

RF PARTS™ CO.

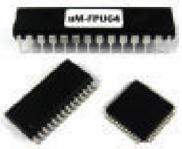
Tubes, Transistors, Power Components

Email: rfp@rfparts.com • Web: www.rfparts.com

800-737-2787 Fax 888-744-1943

uM-FPU64

64-bit Floating Point Coprocessor



DIP-28, SOIC-28,TQFP-44

64-bit and 32-bit IEEE 754 compatible
SPI or I2C interface, 3.3V Supply, 5V Tolerant I/O
Extensive floating point support, FFT, matrix operations
GPS input, local device support, Flash functions, IDE

Robotics
Navigation
Sensor Modules
Embedded Systems

Additional products...
uM-FPU V3.1 32-bit FPU
uM-PWM1 Servo Coprocessor

www.micromegacorp.com

SURPLUS

SURPLUS ELECTRONIC PARTS & ACCESSORIES

Over 20,000 Items in Stock.

Cables, Connectors, Displays, Fans
Hardware, LEDs, Motors, Potentiometers
Relays, Semiconductors, Service Manuals, Speakers
Switches, Test Equipment, Tools, VCR Parts

Surplus Material Components
SMC ELECTRONICS
www.smcelectronics.com

No Minimum Order.
Credit Cards and PAYPAL Accepted.
Flat $4.95 per order USA Shipping.

KIT BUILDING

KIT BUILDERS

Soldering and Assembly Service
Surface Mount or Thru-Hole
Components for one part, a selection of parts, the entire assembly or production quantities

408-842-8264 Work 831-578-5206 Cell
Wa6ouw@aol.com
www.kitbuildersqrpassemblyservice.com

ENERGY SOLUTIONS

Save Money On Your Electric Bill

www.sunny-solutions-shop.com

800-315-0309

www.servomagazine.com

MOBILE APP NOW AVAILABLE!

SERVO

Download NOW On Your Favorite Mobile Device!

FOR ROBOT BUILDERS

iOS • ANDROID • KINDLE FIRE

LIGHTING

Automatic Stair Lighting

www.ReactiveLighting.com

HARDWARE WANTED

DEC EQUIPMENT WANTED!!!

Digital Equipment Corp.
and compatibles.
Buy - Sell - Trade

CALL KEYWAYS 937-847-2300
or email buyer@keyways.com

READER-TO-READER TECHFORUM

All New TECHFORUM Online At www.nutsvolts.com/tech-forum

>>> QUESTIONS

Voltage Filter

I have a 40 year old Chevy. The fuel gauge bounces half the gauge reading when it gets under a 3/4 tank. I put in a new sending unit but that did not help. I tried a digital gauge and that helped some, but not enough. It is a 90 ohm sending unit, zero ohms (grounded) = empty and 90 ohms = full. I would like to find a circuit that will filter out the high and low spikes, and put out an average voltage reading based on the sending unit's position.

#7131 **Philip Diedeman**
Phoenix, MD

Tech Jobs

I'm trying to get my 18 year old interested in a technical career. I need opinions as to what skills are the most marketable and have the best chance of employment in today's reality. Are online or off campus trade schools worth the money, or do companies favor traditional college degrees?

#7132 **Randy Sutton**
Chicago, Il

Battery Damage?

I have eight six volt lead acid batteries that provide "house" power for my boat. The batteries have run dry on several occasions. It's hard to tell if the "run time" has been adversely affected since I don't usually allow them to fully discharge. Have they likely been damaged, and if so, will equalizing them restore lost capacity?

#7133 **Sam Garberich**
Myrtle Beach, SC

>>> ANSWERS

[#4133 - April 2013]

I'm trying to use an SDX01G2 pressure sensor to log CPAP breathing patterns. The circuit uses an LM324N op-amp, an ADC0831 CCN eight-bit A/D converter, and a BASIC Stamp 2 processor. The problem is the sensitivity. The output goes from 0 up to 191, and then back to 0. I have tried several op-amp configurations with little success. I am a newbee, and could sure use some help.

The SDX01G2's datasheet shows the specs to be rather narrow.

Consider the Freescale MPXV5004 or similar. Its range (in and out) is far wider than the SensorTechnics model. The 5004 was used in my CPAP project simply because it was a pull from an old CPAP machine.

Terry
via email

[#5132 - May 2013]
Wireless Pushbutton

I built a Jeopardy game console for use in a classroom setting. The issue I have is that the player buttons are attached to the console via cables and this gets cumbersome in class. I'd like to find a wireless solution. I've looked at commercial solutions and they are all north of $200. Can anyone suggest a circuit that I could use that would be relatively inexpensive and easy to build?

#1 A solution that worked for me several years ago was to use a wireless doorbell. These generally have a jumper that you can move to avoid interfering with a neighbor that has the same model. You would have to purchase four units and then set each one to a different "channel."

If your console is not made of metal, you could mount the base units inside the console. Disconnect the speakers if you do not want them to chime.

A quick check on Google shows these are available in the $10 to $15 price range. The Honeywell RCWL105A is one that might work for you. There are both 120 VAC and battery operated base units. The pushbuttons are battery operated in either case.

The advantage of the 120 VAC units is that you do not have to purchase batteries and also remember to remove them when not in use. The disadvantage is it may be more difficult to interface these to your console. I used an AC powered base and added a small 5V relay to provide a contact closure to operate my garage door opener. Note that this will be a momentary contact closure.

I could not find a schematic for the unit I used, but was able to use an oscilloscope to find a pin that changed state when I pushed the button. The relay takes too much current to be driven directly, so I added a FET to act as a low side driver. An optical isolator might also work instead of the relay. These can be driven with only a few mA if the output transistor only has to sink a few mA.

***BE VERY CAREFUL** and don't even attempt to connect an oscilloscope to*

Send all questions and answers by email to **forum@nutsvolts.com** or via the online form at **www.nutsvolts.com/tech-forum**

All questions *AND* answers are submitted by *Nuts & Volts* readers and are intended to promote the exchange of ideas and provide assistance for solving technical problems. All submissions are subject to editing and will be published on a space available basis if deemed suitable by the publisher. Answers are submitted by readers and **NO GUARANTEES WHATSOEVER** are made by the publisher. The implementation of any answer printed in this column may require varying degrees of technical experience and should only be attempted by qualified individuals.

Always use common sense and good judgment!

an AC powered unit unless you have an isolation transformer, since these units may operate directly from the120 VAC with no isolation. The relay or optical isolator will provide the isolation once you have found a place to connect it. You can contact me via email at **m.e.whitmore@comcast.net** if you have questions.

Mark E. Whitmore
Frederick, MD

#2 For wireless control, it is tough to beat the low cost of Sony TV remotes from the local Goodwill, teamed with the IR-D15A 15-bit Sony® IR decoder IC from Reynolds Electronics. For RF remote control, there are multiple articles on how to use PIC or Atmel (Arduino) processors using serial communication; simply search or stop by the Microchip site.

An alternate encode/decode is the TX-2/RX-2 "R/C car" remote control IC sets available from eBay. I recommend the TX-3/RX-3, and buy in pairs. If using RF instead of infrared, consider the WRL-10535 and WRL-10533 transmitter/receivers from **www.sparkfun.com**, which I personally use with the TX-2/RX-2 R/C car set for fan control. The TX-2 can connect directly to an RX-2 for debugging, and it is easy to scope the output. A PIC (16F88) or ATmega32x (Arduino) will probably help for "who clicked first." Holtek also produces inexpensive encoder/decoder pairs; **www.allelectronics.com** has Sharp IR decoders which work fine with the Sony remotes, AVRs, and much more.

Jim Lacenski
Bellevue, WA

[#6131 - June 2013]
Low Power IR Remote

I would like to make an IR remote control to be used in nursing homes to activate alarm equipment. Specifications are: Receiver: nine volt, VERY LOW power consumption; Transmitter: handheld, battery powered, three volt (lithium?), 2-5 meter range.

Regarding low power remotes, check out Linx Technologies at **https://www.linxtechnologies.com/en/products/remotes** and **https://www.linxtechnologies.com/aen/products/modules**.

They provide receivers and keyfob-type transmitters.

Peter Goodwin
Rockport, MA

[#6132 - June 2013]
Project Building

What's the best way to cut a one inch hole in a metal or plastic project box? What about a square hole?

#1 The best way to cut a one inch hole in plastic or metal boxes is to use a chassis punch. Greenlee makes such punches. A one inch punch is available from Grainger (see **www.grainger.com/Grainger/GREENLEE-Round-Knockout-Punch-4A738?Pid=search**), as well as many electrical distributors.

Problem is that they're expensive because they're meant for cutting through fairly thick steel. The 1" device from Grainger sells for about $87.

Best bet is to look on eBay for an antique radio chassis punch. At this writing, I found a one inch punch offered for under $10 *(plus shipping)*.

Peter Goodwin
Rockport, MA

#2 While it's virtually impossible to declare a "best" way to do any number of things, I'll share a couple of my favorites.

For cutting round holes in thin materials, it's hard to beat a step drill bit. These are cone shaped cutters that fit in an ordinary drill or drill press, and do a great job of cutting or enlarging holes up to about 1.5 inches without mangling the material.

For square holes, I often use a nibbler tool, following up with a file to clean and square up the cut. With a bit of practice, you can use the nibbler to cut all sorts of shapes.

James Sweet
via email

[#6133 - June 2013]

Drive By Wire

I want to build a 'drive by wire' system for a project that I have been working on. I want to operate the throttle on a 14 HP engine via a servo with the gas pedal hooked to a potentiometer, so that when the 'gas' pedal is pushed, it will open the throttle on the engine and vice versa (close when 'gas' pedal is released). The servo would need to return to the same position as the pedal potentiometer.

The circuit needs to run on a 10-15 volt DC automotive voltage, and be turned on and off with the ignition switch. The servo must be able to hold the throttle position for as long as the driver has his foot on the gas pedal in any position, and then return to the zero position when the foot pedal is released. Can someone help me accomplish this?

A stepper motor should work; there are already stepper motors being used in cars under the hood. You will need:

- An encoder to measure the position of the gas pedal.
- A device to convert this position into pulses.
- Another device to use the pulses to power a stepper motor.
- And, of course, the stepper motor itself.

Digi-Key sells an encoder with 16 different pulse counts per revolution, set with tiny switches for $26; part # AMT103-V. US Digital sells a chip that will convert the encoder output into pulses that a stepper driver can use; part # LFLS7184, $3.50. Interinar sells a stepper driver board for $36. Stepper motors are available as surplus everywhere.

You will have to have an idea of which stepper you need if you decide to go this route. You will also need a five volt power supply for the encoder and the 7184 chip. US Digital also sells encoders but none that have 16 different resolutions, as far as I know. I have used all the devices mentioned, bought from the companies mentioned. I have used them for a similar application, so I know they will work. The encoder mentioned is a robust device that will be better over time than a cheap pot to measure the foot pedal position. Since the encoder has so many different resolutions, it should be fairly easy to find a ratio between the foot pedal position and the throttle opening.

You could also use an RC servo to open the throttle. These servos need a pulse train of a particular width to know how much to move. The pulses must be constant because when the pulse train stops, the servo returns to the starting position. So, you need some kind of servo controller that will use a pot connected to the foot pedal to tell it how wide to make the pulse it sends to the servo.

The stepper solution suggested above pretty much just requires connecting the different parts with wires according to the info in the datasheets. However, you can make your own servo controller from a 555 timer, some fixed resistors, some variable resistors (a.k.a., pots), and some capacitors. You will need to solder all this together in an enclosure. This link: **pcbheaven.com/wikipages/How_RC_Servos_Works/** will explain how RC servos work and has a sample circuit that you can build to control the RC servo.

One advantage of the RC servo is that it is an absolute positioning system. A certain resistance measured at the foot pedal results in a certain throttle position. The stepper system described above does NOT work that way. The throttle must be at the zero position when power is removed. If the throttle is at half way when power is removed, then the pedal will start to move the throttle from the half way position when power is restored — unless some type of mechanism always returns the throttle to the fully closed position when power is removed.

Eric Snow
via email

[#6135 - June 2013]

Salvaging Parts

Since money for hobbies is hard to come by these days, is it worth trying to salvage parts from old electronic equipment to build new projects? What kind of equipment has the most usable parts — old TVs, computers, stereos? How far do component values deteriorate over time?

#1 Absolutely! Since I was a kid, I've kept a stash of boards from dismantled equipment from which to salvage usable parts. While it started out of necessity, even now that I can afford to buy new parts for projects, it's still handy to have a selection of salvageable stuff on hand, and the best form of recycling is reuse.

If you are anything like me, sooner or later it will be 2 am on a Saturday and you will find yourself needing a specific part you don't have and the junk box will save the day.

Most components hold up well with age. Semiconductors are almost always good, unless you are unlucky enough to find the part that resulted in the equipment being discarded in the first place. Reasonably modern components don't tend to drift much, although electrolytic capacitors may be suspect, particularly if they were used in a switchmode power supply.

James Sweet
via email

#2 It is certainly worth salvaging parts from old electronics. My favorite tools for this are a desoldering iron (Parts Express model 372-202) and a Black & Decker heat gun (model HG1300). A good first device to take apart would be a VCR. Look for a heavy one made in the ‘80s or early ‘90s; it will have many basic parts like capacitors and inductors, along with more interesting bits such as a VFD display and a motor control chip.

In terms of what else to take apart: I would not recommend TVs; you can get a nasty shock from one, and the boards don't have much on them. Older computer boards have more through-hole chips than newer ones.

(Heat gun works well here.) Stereos have a lot of good parts, from heavy duty transistors to graphic equalizer chips.

I have not seen much deterioration of component values. Very old electrolytics sometimes can't hold a good charge. Most of the resistors in an old Vidicon camera I took apart were out of spec. In comparison, stuff from 20-30 years ago normally holds up very well. Capacitors with 'J' markings and resistors with gold bands almost always measure as expected.

Also, to dispel a couple of 'myths' about salvage that I frequently hear. “Don't bother removing LEDs or transistors, they always burn up.” This is not true; you can save anything if you work quickly enough. “Remember that salvaging costs money in electricity to run the iron/heat gun.” Not really. At 12 cents per KW/H, it costs about half a penny to use the 45 watt iron for an hour. If I use the heat gun, the cost per hour rises to a measly 16.5 cents.

Nicholas Amrich
via email

### [#6137 - June 2013]
Follow-up To #7121

I switched my tube system to separate units now using a Scott 350B. I need to check multipath distortion as in Tech Forum #7121, but wonder what bandwidth I need for a scope. I assume 100 MHz, but would a lower bandwidth (less costly) scope work okay? Would I hook it up as noted in #7121?

The oscilloscope only needs to be flat to 15 kHz, so almost any one with at least 100 kHz bandwidth will work. I did it with one of the 5 MHz/5" scopes that were readily available since the 1950s (Dumont, Eico, Heathkit, KnightKit, RCA, etc.). You should be able to find a used one for less than $20. The connections are the same: FM detector to the X input and AGC to the Y input.

Joseph Feng
San Jose, CA

■ LOOK FOR ■ SEARCH FOR ■ FIND

Find your favorite advertisers here!

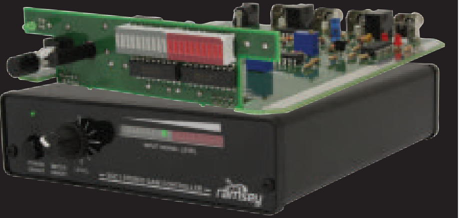
ADvertiser INDEX

Beat The Heat...Build a Kit!

Ramsey Kits Are Always Cool, Even In The Summer Heat!

Stereo Audio Platform Gain Controller

- ✔ Stereo audio processing while preserving audio dynamics!
- ✔ True stereo control keeps virtual sonic source location intact!
- ✔ Automatic set-and-forget operation!
- ✔ Auto-bypass restores original levels when power is turned off!
- ✔ Unbalanced RCA and 3.5mm stereo input and output line level jacks!
- ✔ Built-in bar graph indication of signal level with display mute!



Your Full-Time Audio Engineer... Without the Payroll!

The SGC1 is one of our latest kits, and provides a great solution to the age-old problem: how can we easily correct inconsistent audio levels without negatively affecting the dynamics of the audio signal? The SGC1 circuit implements a principle known as the "Platform Gain Principle," which was originally developed by CBS Labs (what we now know as CBS in the TV and radio world) to allow transmitted audio levels to be automatically adjusted to keep them within a desired range.

Think of it like an audio engineer, constantly adjusting the output level in order to limit highs that would be too loud while boosting lower levels so that they can still be heard. You may think "oh, this is just another limiter/compressor!" Not so! Here's the real trick: keeping the full dynamic range ratio of the output signal the same as the original input - something the typical limiter/compressor can only dream of doing! The SGC1 can be placed in just about any standard analog stereo line level audio circuit (the red and white RCA connectors or the mini-phone connector) to keep the audio level within the desired range. It's also the perfect addition to any of our hobby kit transmitters, allowing you to match levels between different audio sources while keeping lows audible and preventing the highs from overdriving.

The SGC1 makes a great addition to any audio system where you need to keep levels from different sources under control, but still make sure they all sound great! In addition to its useful basic function and great audio performance, the SGC1 also boasts a front panel LED meter to give an indication of the relative level of the input signal, plus a level control (also on the front panel) that allows you to adjust the controller to the min/max center point of your desired level range. And yes, it is a **S**tereo **G**ain **C**ontroller! Meaning that the levels of both the left and right channels are monitored and adjusted equally, thereby maintaining the relative virtual position of things like instruments, singers and speakers! When powered down the controller is automatically bypassed, and will simply pass through the unchanged original audio, meaning no messing around with cables required if you simply want to cut it out of the equation temporarily. There's even a front panel switch to disable the LED display - perfect for situations where light can be distracting - like a darkened home-theatre! The entire unit is housed in a slim attractive black textured aluminum case that is sure to complement your studio or home theatre. If you're looking for perfect audio levels, hire a broadcast audio engineer, but if that doesn't fit your budget, the SGC1 is the next best thing! Includes 15VDC world-wide power adapter.

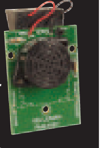
SGC1 Stereo Audio Platform Gain Controller Kit, with Case and Power Supply $179.95

Audio Recorder & Player

Record and playback up to 8 minutes of messages from this little board! Built-in condenser mic plus line input, line & speaker outputs. Adjustable sample rate for recording quality. 4-switch operation that can be remote controlled! Runs on 9-12VDC at 500mA.

K8094 Audio Recorder/Player Kit $32.95

Water Sensor Alarm



This little $7 kit really shined during Sandy! Simply mount the alarm where you want to detect water level problems. When the water level rises it touches the contacts and the alarm goes off! Sensor can even be remotely located. Runs on a 9V battery (not included).

MK108 Water Sensor Alarm Kit $6.95

12VDC Regulated Switching Supply

Go green with our new 12VDC 1A regulated supply. Worldwide input 100-240VAC with a Level-V efficiency! It gets even better, includes DUAL ferrite cores for RF and EMI suppression. All this at a 10 buck old wallwart price!

AC121 12VDC 1A Regulated Supply $9.95

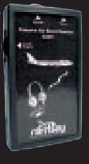
Electronic Watch Dog

A barking dog on a PC board! And you don't have to feed it! Generates 2 different selectable barking dog sounds. Plus a built-in mic senses noise and can be set to bark when it hears it! Adjustable sensitivity! Unlike my Saint, eats 2-8VAC or 9-12VDC, it's not fussy!

K2655 Electronic Watch Dog Kit $39.95

Passive Aircraft Monitor

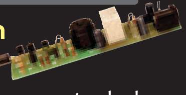
PATENTED!



The hit of the decade! Our patented receiver hears the entire aircraft band without any tuning! Passive design has no LO, therefore can be used on board aircraft! Perfect for airshows, hears the active traffic as it happens! Available kit or factory assembled.

ABM1 Passive Aircraft Receiver Kit $89.95

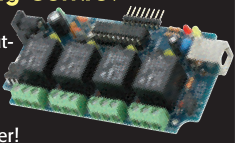
Laser Trip Sensor Alarm



True laser protects over 500 yards! At last within the reach of the hobbyist, this neat kit uses a standard laser pointer (included) to provide both audible and visual alert of a broken path. 5A relay makes it simple to interface! Breakaway board to separate sections.

LTS1 Laser Trip Sensor Alarm Kit $29.95

4-Channel USB Relay Control



This professional quality USB relay controller allows computer controlled switching of external devices, plus full bi-directional communication with the external world using the USB port of your computer!

The controller features four onboard relay outputs with a current rating of 10A each. Also onboard is a 6-channel Input/Output interface, with each channel individually configurable as Digital Input, Digital Output, Analog Input (10-bit Resolution). In Digital Input/Output modes, each channel can support a TTL compatible or ST input or a 5V output signal. In Analog Input mode, each channel can convert a voltage of between 0 to 5V into a 10-bit digital representation.

UK1104 4-Ch USB Relay Interface Kit $59.95

The Learning Center!

Beginners To Advanced... *It's Fun!*

- ✔ Learn, build, and enjoy!
- ✔ 130, 200, 300, & 500 in one electronic labs!
- ✔ Practical through hole and SMT soldering labs!
- ✔ Integrated circuit AM/FM radio lab!
- ✔ AM, AM/FM, and SWL radio labs!
- ✔ Radio Controlled (RC) car!
- ✔ Beginner's non-soldering kits!

For over 4 decades we've become famous for making electronics fun, while at the same time making it a great learning experience. As technology has changed over these years, we have continued that goal!

PL130A Gives you 130 different electronic projects together with a comprehensive learning manual describing the theory behind all the projects.

PL200 Includes 200 very creative fun projects and includes a neat interactive front panel with 2 controls, speaker, LED display and a meter.

PL300 Jump up to 300 separate projects that start walking you through the learning phase of digital electronics.

PL500 The ultimate electronics lab that includes 500 separate projects that cover it all, from the basics all the way to digital programming.

SP1A Whether young or old, there's always a need to hone your soldering skills. Either learn from scratch or consider it a refresher, and end up with a neat little project when you're done!

SM200K Move up to Surface Mount Technology (SMT) soldering, and learn exactly how to solder those tiny little components to a board!

AMFM108K We not only take you through AM and FM radio theory but we guide you through IC's. When you're done you've built yourself an IC based AM/FM radio that works great!

AM2 Learn the complete theory of AM broadcast radio and end up with a highly sensitive AM radio receiver!

AMFM7 Step up to AM/FM with this multi-band lab and learn the basics of both bands. The FM tuner is factory assembled and aligned!

SR3 Enter the world of SWL with this shortwave radio learning lab covering 6-8MHz and 12-18MHz!

AK870 One of the most exciting electronic learning kits that the kids will love! Build a complete RC speedster from the ground up! 7 remote functions!

PL130A	**130-In-One Lab Kit**	**$39.95**
PL200	**200-In-One Lab Kit**	**$84.95**
PL300	**300-In-One Lab Kit**	**$109.95**
PL500	**500-In-One Lab Kit**	**$249.95**
SP1A	**Through Hole Soldering Lab**	**$9.95**
SM200K	**SMT Practical Soldering Lab**	**$22.95**
AMFM108K	**AM/FM IC Lab Kit & Course**	**$36.95**
AM2	**AM Radio Learning Lab**	**$11.95**
AMFM7	**AM/FM Radio Learning Lab**	**$12.95**
SR3	**Short Wave Radio Learning Lab**	**$16.95**
AK870	**RC Speedster Car Kit**	**$29.95**

ramsey
www.ramseykits.com

Where Electronics... Is Always Fun!

- ✔ *Build It!*
- ✔ *Learn It!*
- ✔ *Achieve It!*
- ✔ *Enjoy It!*

Follow Us and SAVE $$

Follow us on your favorite network site and look for a lot of super deals posted frequently... exclusively for our followers!

Next Generation Classic Nixie Tube Clocks!

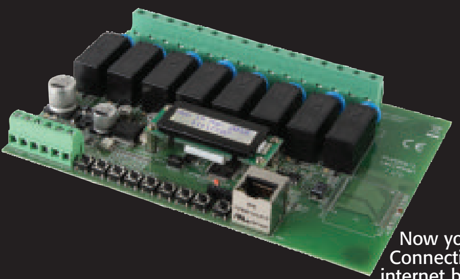
- ✔ Our brand-new next generation clock for 2013!
- ✔ Today's latest technology with yesterday's display!
- ✔ Choice of hand rubbed Teak/Maple, polished aluminum, or acrylic base!
- ✔ Choice of Nixie tubes types
- ✔ Low-cost GPS receiver timebase option!
- ✔ Programmable color LED tube mood lighting!
- ✔ The ultimate conversation piece!

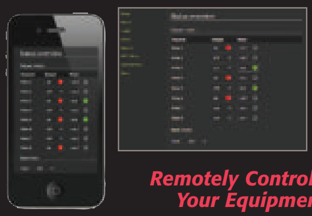
NEW FOR 2013!
Our next generation of classic Nixie tube clocks perfectly mesh today's technology with the Nixie era technology of the 60's. Of course, features you'd expect with a typical clock are all supported with the Nixie clock... and a whole lot more! Time wise, the clocks are designed around a quartz crystal timebase, therefore are not AC power frequency dependent like a lot of clocks. This means they can be used in any country regardless of power frequency, with the included 12VDC regulated power supply. The clocks are also programmable for 12 or 24 hour mode, various AM/PM indications, programmable leading zero blanking, and include a programmable alarm with snooze.

Unlike most Nixie clocks, the clocks also display the date in DD.MM.YY, MM.DD.YY, or YY.MM.DD format, which can be programmed to display for a few seconds at the end of each minute either as a static display, or by a neat scrolling in/out from alternating sides of the display. Display wise, the clocks feature a programmable night mode with dim or blank display, a programmable master blank tube saver, hard or soft fade digit change, and even have a built-in "Slot Machine" cathode poisoning prevention routine. Programming and setting the clock is a breeze with simple 2-button entries on the rear panel. The clocks are available in our signature hand rubbed Teak & Maple, polished aluminum, or clear acrylic bases.

WE CROSSED THE TECHNOLOGY TIMELINE!
We then jumped the technological time line of the 60's Nixie displays by adding the latest multi-colored LEDs to the base of the Nixie tubes to provide hundreds of illumination colors to highlight the glass tubes! The LED lighting can be programmed to any color and brightness combination of the colors red, green, or blue to suit your mood or environment. Then we leaped over the technological timeline by integrating an optional GPS time base reference for the ultimate in clock accuracy! The small optional GPS receiver module is factory assembled and tested, and plugs directly into the back of the clock to give your Nixie clock accuracy you could only dream of! The new series clocks are available in 6-tube and 4-tube versions, with your choice of bases, and your choice of kit or factory assembled & tested. If you're looking for the ultimate conversation piece, with a trip down nostalgia lane, check out our clocks at www.ramseykits.com/nixie.

NIXIE CLOCKS **Classic Nixie Tube Clocks, Teak/Maple, Polished Aluminum, or See-Through Acrylic Base, Kit or Factory Assembled** **From $229.95**





Remotely Control and Monitor Your Equipment On-Line!

8-Channel Ethernet Remote Controller

- ✔ 8 separate relay controls!
- ✔ Built-in web interface for easy operation!
- ✔ User defined TCP server port!
- ✔ Static or DHCP IP address!
- ✔ Sends email at power-up and event changes!
- ✔ Multiple user authentications
- ✔ On-board LCD display

Now you can easily control and monitor up to 8 separate circuits via the standard Ethernet network in your home or office. Connection wise it couldn't be simpler. The controller functions as an IP based web server, so it can be controlled by any internet browser that can reach your network! There are no drivers or proprietary software required, just access the controller like any web page from your PC, laptop, or even your smartphone! Security is assured allowing up to 4 separate user credentials. The controller can be set to a specific static IP within your network subnet or can be set to DHCP (auto negotiate). The controller can even be programmed to send you an email to notify and confirm power up and status changes!

To simplify the connection of your equipment to the controller, 8 separate and isolated relay outputs are provided! This gives you internet or network control of up to 8 separate functions. No need to worry about interfacing a logic high or logic low, or burning up the interface! The applications are endless! From something as simple as turning on and monitoring lights at your house with a normal latched closure to advanced control of your electronic gadgets, radio equipment, or even your garage door! Each relay contact is rated at 12A at 30VDC or 16A at 230VAC. Each of the 8 channels has built-in timer and scheduler programs for day, weekend, working days, every day, and every day except Sunday. Relay control functions are programmable for on, off, or pulse (1-99 seconds, 1-99 minutes, or 1-99 hours). In addition to control functions, the web interface also displays and confirms the status of each channel. Each channel can be custom labeled to your specific function name. The controller operates on 12VDC or 12VAC at 500mA or our new AC121 global 12VDC switching power supply over to the left! Factory assembled, tested, and ready to go! Even includes a Cat-5 cable!

SPECIFICATIONS

Channels:	8, separate relay outputs
Format:	IEEE 802.3 compatible Ethernet, 10Mbps
TCP/IP:	IPV4, static or DHCP
NETBios:	User defined name
Server:	TCP, SNTP, email
TCP Port:	User defined
Display:	On-board LCD, time or network status
Labels:	User defined for input and all outputs
Users:	Administrator plus 3 users
Timers:	28 times max
Contacts:	12A@30VDC, 16A@230VAC x 8
Power:	12VDC, 12VAC, 500mA
Mounting:	PC board or optional DIN rail enclosure

VM201 **8-Channel Remote Ethernet Controller, Factory Assembled & Tested** **$169.95**

GET THE NUTS & VOLTS DISCOUNT!

*Mention or enter the coupon code **NVRMZ12** and receive 10% off your order!*

800-446-2295
www.ramseykits.com

Prices, availability, and specifications are subject to change. We are not responsible for typos, stupids, printer's bleed, or 4th of July fireworks accidents! In the true holiday spirit, Robin made me do this ad copy instead of going out on the boat! Not Fair! Visit www.ramseykits.com for the latest pricing, specials, terms and conditions. Copyright 2013 Ramsey Electronics®...so there!

RAMSEY ELECTRONICS®
590 Fishers Station Drive
Victor, NY 14564
(800) 446-2295
(585) 924-4560

CAD CONNECTED

Proteus Version 8
Launch Offer - SAVE 10%
Use NVL2013ATJ
When placing your order

PROTEUS DESIGN SUITE VERSION 8

Featuring a brand new application framework, common parts database, live netlist and 3D visualisation, a built in debugging environment and a WYSIWYG Bill of Materials module, Proteus 8 is our most integrated and easy to use design system ever. Other features include:

- Hardware Accelerated Performance.
- Unique Thru-View™ Board Transparency.
- Over 35k Schematic & PCB library parts.
- Integrated Shape Based Auto-router.
- Flexible Design Rule Management.
- Polygonal and Split Power Plane Support.
- Board Autoplacement & Gateswap Optimiser.
- Direct CADCAM, ODB++, IDF & PDF Output.
- Integrated 3D Viewer with 3DS and DXF export.
- Mixed Mode SPICE Simulation Engine.
- Co-Simulation of PIC, AVR, 8051 and ARM MCUs.
- Direct Technical Support at no additional cost.

labcenter
Electronics
www.labcenter.com

Visit our website or phone 866.499.8184 for more details

Labcenter Electronics North America 411 Queen St, Newmarket, Ont. Canada L3Y 2G9
Email: info@labcenter-electronics.com Tel: 905.898.0665 Fax 905.898.0683